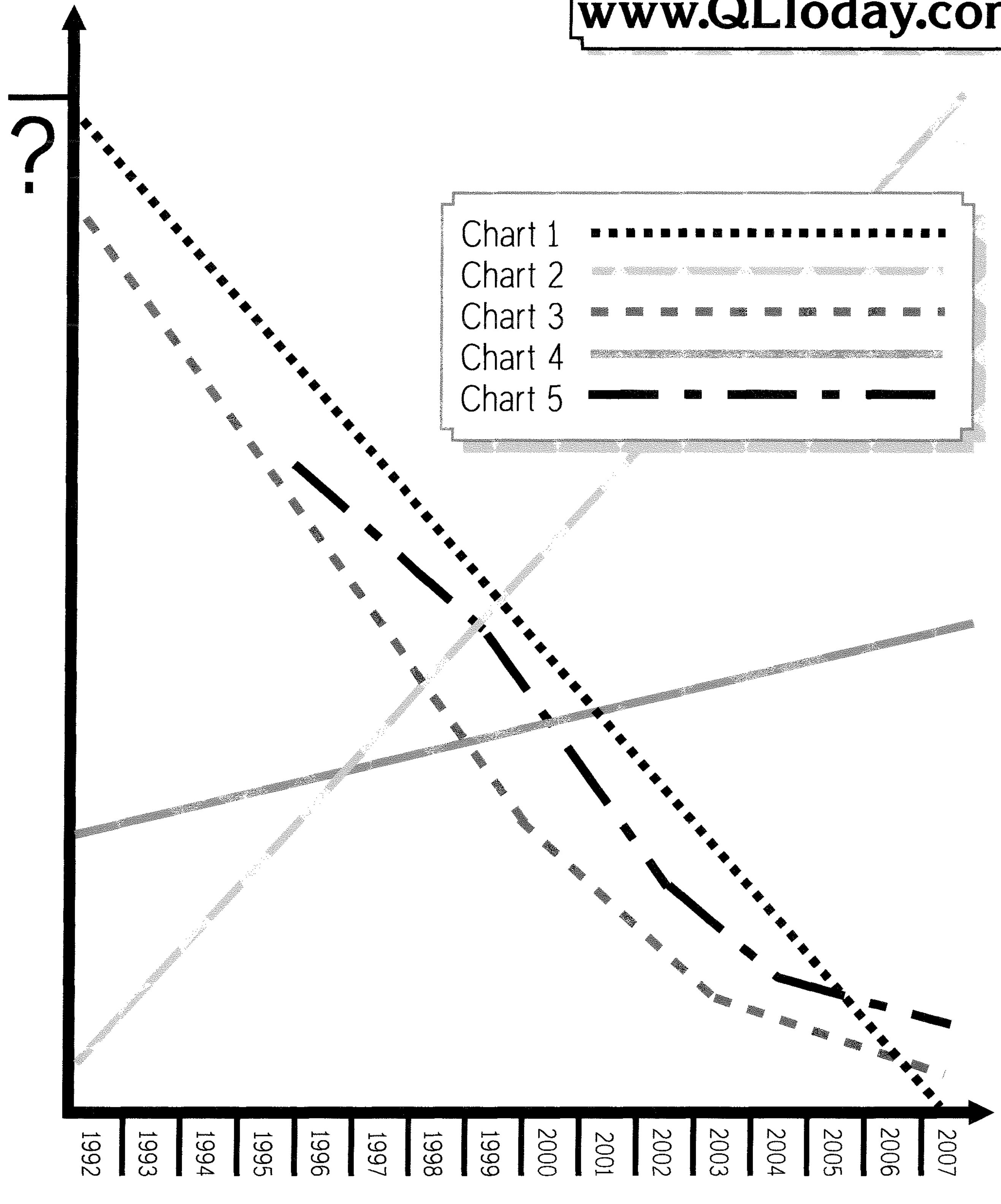
QL Today

Volume 11
Issue 1
July/August
2006

ISSN 1432-5454

The Magazine about QL, QDOS, Sinclair Computers, SMSQ...

www.QLToday.com



Contents

Advertisers

in alphabetical order

QL *Today*

ISSN 1432-5454

German office & Publisher:

Jochen Merz Software	Tel. +49 203 502011
Kaiser-Wilhelm-Str. 302	Fax +49 203 502012
47169 Duisburg	email: JMerz@j-m-s.com
Germany	email: QLToday@j-m-s.com

English office:

Q Branch	Tel. +44 1273 386030
20 Locks Hill	Mobile +44 7836 745501
Portslade	Fax +44 1273 381577
BN41 2LB	email: qbranch@qbranch.demon.co.uk
United Kingdom	email: QLToday@j-m-s.com

Editor:

Geoff Wicks	Tel. +44 1332 271366
5b Wordsworth Avenue	email: gwicks@beeb.net
Sinfin	email: QLToday@j-m-s.com
Derby DE24 9HQ	
United Kingdom	

Co-Editor:

Bruce Nicholls	Tel +44 20 71930539
38 Derham Gardens	Fax +44 870 0568755
Upminster	email: qltoday@q-v-d.demon.co.uk
Essex RM14 3HA	email: QLToday@j-m-s.com
United Kingdom	

QL *Today* is published five times a year, our volume begins on beginning of June. Please contact the German or English office for current subscription rates or visit our homepage www.QLTODAY.com.

We welcome your comments, suggestions and articles. YOU make **QL *Today*** possible. We are constantly changing and adjusting to meet your needs and requirements. Articles for publication should be on a 3.5" disk (DD or HD) or sent via Email. We prefer ASCII, Quill or text87 format. Pictures may be in _SCR format, we can also handle GIF or TIF or JPG. To enhance your article you may wish to include Saved Screen dumps. PLEASE send a hardcopy of all screens to be included. Don't forget to specify where in the text you would like the screen placed.

The deadline for the next issue is the 30th of September 2006

Editorial

by Geoff Wicks

What better way to begin a new volume of QL Today than with a challenge? And begin means just that, because the challenge is on our cover. We believe our readers are creative people, and we would like you to use your imagination to suggest what this cover tells us about QL development.

On the x-scale are 16 years starting at 1992 and ending at 2007. The y-scale is a bigger problem because we want you to tell us what this scale represents.

Then there are five different graphs. Four begin in 1992, but the fifth in 1997. What happened in that year that was so special? Maybe you will have to go back to volume 4 issue 6 and see what QL historian, Dilwyn Jones, had to say about that year in his QL family tree article. Or, if you are a more thorough historian who prefers original sources, go right back to the start of QL Today and examine the content of volumes 1 and 2. And if you need a little help Brian Kemmett has now released a PDF version of the index to all 10 volumes of QL Today.

To start you off we have a number of suggestions:

- "New software developments"
- "Number of QL Today readers"
- "Visitors to QL shows"
- "Dealer's travelling costs per customer"
- "Quanta's bank balance"
- "Number of original black boxes still in use"

You can, of course, make it as serious or as facetious as you like. All contributions are welcome.

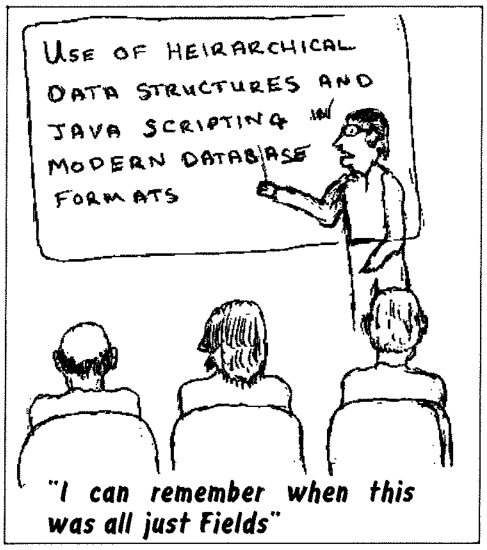
To some this may seem like a silly game, but there is a serious intention. It is an unorthodox way of trying to get a picture of existing QL users and of QL Today readership. Sometimes an open-ended approach can deliver more than a straightforward factual questionnaire.

One thing is certain. The number of QL-ers may be in decline, but the liveliness of the community remains. Summer is traditionally a quiet QL time and the QL users email list has been quieter than usual this summer. But behind the scenes a lot of work has been going on, particularly on the software front, as can be seen from our news section. There has also been a welcome diversity in the topics of the articles we have received.

All in all an encouraging time to begin a new volume of QL Today. Let's hope we can maintain the momentum and can be just as optimistic in a year's time.

Cartoon

by Roy Wood


USE OF HEIRARCHICAL DATA STRUCTURES AND JAVA SCRIPTING IN MODERN DATABASE FORMATS

"I can remember when this was all just Fields"

QSTRIPPER

As promised Norman Dunbar has released his latest version of QStripper which can convert Quill files into PDF and other formats. In his own words:

Right then, as promised, QStripper has been put up on the Web for a severe testing session. I've done my bit and found a couple of bugs (details later) but nothing too 'heavy'. It's your turn now! Until I have access to my Linux system with QT4 (it's running QT3 at the moment) I've not got a set up running on Linux yet, however, watch this space.

The place to go is here:

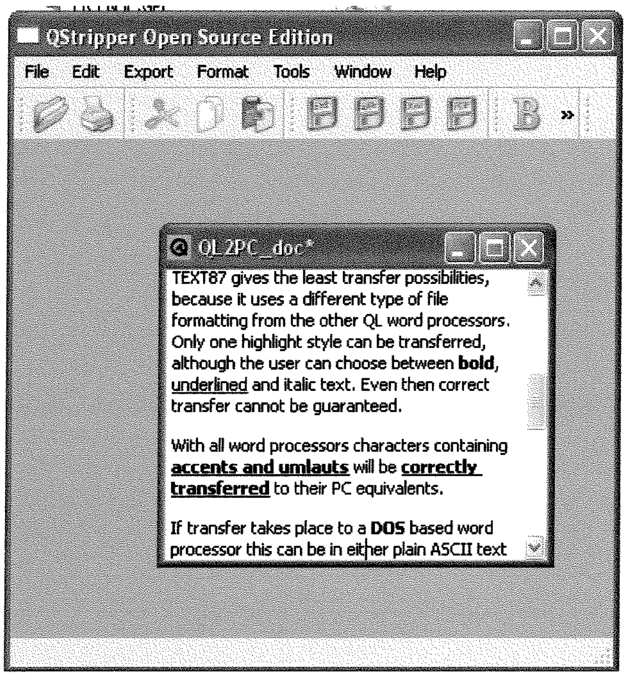
http://www.bountiful.demon.co.uk/qstripper.html

Although QStripper itself has a file length of only 70Kb and its source code 90Kb, it is also necessary to download the Windows support files, which are a massive 3Mb. (Qstripper makes use of the Open source version of Trolltech's QT4 toolkit for C++ developers.) Norman has usefully put separate download files on the web site to allow for quick downloads when he updates the program.

QStripper is simple to use. You just save the Quill file to a PC medium and then load it into the program. Bold, Underlining, Subscript and Superscript are all preserved and accented characters correctly transferred to their Windows equivalent. Some simple editing of the file is possible.

The file can then be saved as plain text, HTML, DocBook XML and PDF.

Unfortunately Text87 users, who were hoping for an equivalent program are likely to be disappointed. Perfection and Quill have similar formatting with control codes for font changes embedded in the text. Text87 formatting closely resembles that of a PC word processor. Each font change is signalled by chr$(27) and somehow recorded in the formatting information at the end of a file. The detailed formatting of a Text87 file has never been published and so far has defeated the efforts of the experts to unravel it.


QStripper Open Source Edition
File Edit Export Format Tools Window Help
QL2PC_doc*

TEXT87 gives the least transfer possibilities, because it uses a different type of file formatting from the other QL word processors. Only one highlight style can be transferred, although the user can choose between **bold**, underlined and italic text. Even then correct transfer cannot be guaranteed.

With all word processors characters containing ***accents and umlauts*** will be ***correctly transferred*** to their PC equivalents.

If transfer takes place to a **DOS** based word processor this can be in either plain ASCII text

SUQCESS Version 2.04 Update

Bob Spelten writes:

"The last Eindhoven show (June 2006) saw the launch of yet another update to Suqcess.

The most visible change is in the little Hint windows that pop up when the cursor lingers over an item (thanks to Per Witte & QL Today)

Normally Suqcess asks each time how to open the database but that was not appreciated by everyone so now you can set a default in Preferences. However you can override this temporarily in the Command menu. A new database can be made more quickly by reading the field structure from an existing database. If the field types of the open database match another one all records can be merged but you can also merge single records.

Copying a record to scrap or a file will no longer cut the fields at the column width like Print but will take the whole field.

Copying data from field to field has been improved. A text that starts with a date will be converted to a Long INT date field. 01-01-2006 or 1/1/2006 or 1.1.2006 or any combination thereof will be accepted.

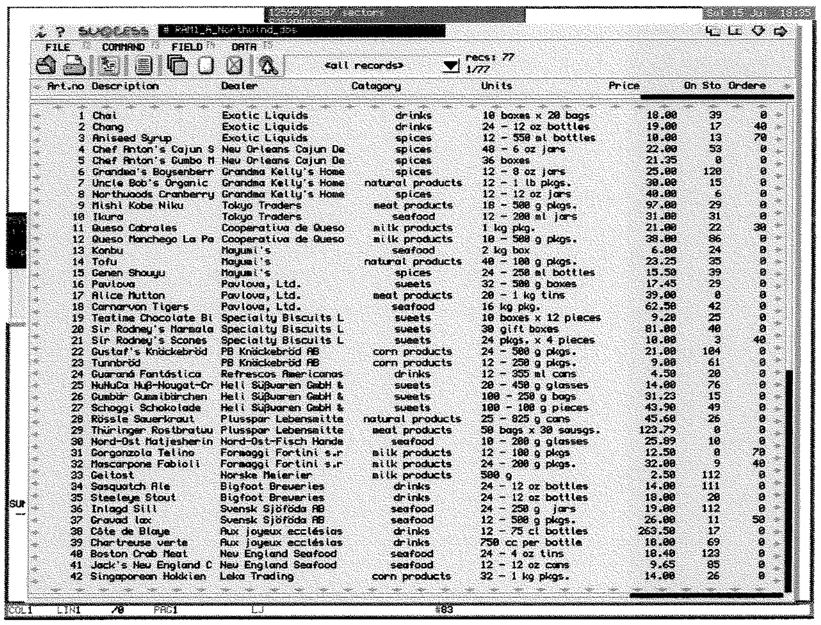
When converting text to a number Suqcess will treat a comma as a decimal point so only one comma or dot is valid. A leading '+' or '-' sign can also be used.

Can you handle more then 100 User-defined searches? Suqcess can now store them all.

The Find/Replace option will create a list of the found records and this will also show up to 60 characters of the field data. From the list you can jump straight to the record.

If your system correctly recognises a Read-Only medium you can open a database from there. But beware, check this before trying because my QXL and my Aurora/Qubide both failed the test and this can crash Suqcess or even your system.

If Suqcess is not configured for a valid directory it will look for the new HOME directory. When this does not exist you will need to configure Suqcess properly.
More details can be found in the Readme file. There I also updated the list of potential conflicts (thanks to the Keywords_dbs from Alex Wells & Fran ois van Emelen) and report on other problems and bug fixes.
Suqcess2 still only runs under SMSQ/E 3. A new attempt was made to have it run under QDOS but I could not get a stable QDOS system running with the latest PE v2 so any testing was doomed to fail.
A trial version can be downloaded from Wolfgang Uhlig's site:
www.uhlich.nl/ql/
Full versions can be obtained from Jochen Merz Software or Q Branch in English, German or Dutch."



Art.no	Description	Dealer	Catagory	Units	Price	On Sto	Ordere
1	Chai	Exotic Liquids	drinks	10 boxes x 20 bags	18.00	39	0
2	Chang	Exotic Liquids	drinks	24 - 12 oz bottles	19.00	17	40
3	Aniseed Syrup	Exotic Liquids	spices	12 - 550 ml bottles	10.00	13	70
4	Chef Anton's Cajun S	Neu Orleans Cajun De	spices	48 - 6 oz jars	22.00	53	0
5	Chef Anton's Gumbo M	Neu Orleans Cajun De	spices	36 boxes	21.35	0	0
6	Grandma's Boysenberr	Grandma Kelly's Home	spices	12 - 8 oz jars	25.00	120	0
7	Uncle Bob's Organic	Grandma Kelly's Home	natural products	12 - 1 lb pkgs.	30.00	15	0
8	Northwoods Cranberry	Grandma Kelly's Home	spices	12 - 12 oz jars	40.00	6	0
9	Mishi Kobe Niku	Tokyo Traders	meat products	18 - 500 g pkgs.	97.00	29	0
10	Ikura	Tokyo Traders	seafood	12 - 200 ml jars	31.00	31	0
11	Queso Cabrales	Cooperativa de Queso	milk products	1 kg pkg.	21.00	22	30
12	Queso Manchego La Pa	Cooperativa de Queso	milk products	10 - 500 g pkgs.	38.00	86	0
13	Konbu	Mayumi's	seafood	2 kg box	6.00	24	0
14	Tofu	Mayumi's	natural products	40 - 100 g pkgs.	23.25	35	0
15	Genen Shouyu	Mayumi's	spices	24 - 250 ml bottles	15.50	39	0
16	Pavlova	Pavlova, Ltd.	sweets	32 - 500 g boxes	17.45	29	0
17	Alice Mutton	Pavlova, Ltd.	meat products	20 - 1 kg tins	39.00	0	0
18	Carnarvon Tigers	Pavlova, Ltd.	seafood	16 kg pkg.	62.50	42	0
19	Teatime Chocolate Bi	Specialty Biscuits L	sweets	10 boxes x 12 pieces	9.20	25	0
20	Sir Rodney's Marmala	Specialty Biscuits L	sweets	30 gift boxes	81.00	40	0
21	Sir Rodney's Scones	Specialty Biscuits L	sweets	24 pkgs. x 4 pieces	10.00	3	40
22	Gustaf's Knäckebröd	PB Knäckebröd AB	corn products	24 - 500 g pkgs.	21.00	104	0
23	Tunnbröd	PB Knäckebröd AB	corn products	12 - 250 g pkgs.	9.00	61	0
24	Guaraná Fantástica	Refrescos Americanas	drinks	12 - 355 ml cans	4.50	20	0
25	NuNuCa Nuß-Nougat-Cr	Heli Süßwaren GmbH &	sweets	20 - 450 g glasses	14.00	76	0
26	Gumbär Gummibärchen	Heli Süßwaren GmbH &	sweets	100 - 250 g bags	31.23	15	0
27	Schoggi Schokolade	Heli Süßwaren GmbH &	sweets	100 - 100 g pieces	43.90	49	0
28	Rössle Sauerkraut	Plusspar Lebensmitte	natural products	25 - 825 g cans	45.60	26	0
29	Thüringer Rostbratwu	Plusspar Lebensmitte	meat products	50 bags x 30 sausgs.	123.79	0	0
30	Nord-Ost Matjesherin	Nord-Ost-Fisch Hande	seafood	10 - 200 g glasses	25.89	10	0
31	Gorgonzola Telino	Formaggi Fortini s.r	milk products	12 - 100 g pkgs	12.50	0	70
32	Mascarpone Fabioli	Formaggi Fortini s.r	milk products	24 - 200 g pkgs.	32.00	9	40
33	Geitost	Norske Meierier	milk products	500 g	2.50	112	0
34	Sasquatch Ale	Bigfoot Breweries	drinks	24 - 12 oz bottles	14.00	111	0
35	Steeleye Stout	Bigfoot Breweries	drinks	24 - 12 oz bottles	18.00	20	0
36	Inlagd Sill	Svensk Sjöföda AB	seafood	24 - 250 g jars	19.00	112	0
37	Gravad lax	Svensk Sjöföda AB	seafood	12 - 500 g pkgs.	26.00	11	50
38	Côte de Blaye	Aux joyeux ecclésias	drinks	12 - 75 cl bottles	263.50	17	0
39	Chartreuse verte	Aux joyeux ecclésias	drinks	750 cc per bottle	18.00	69	0
40	Boston Crab Meat	New England Seafood	seafood	24 - 4 oz tins	18.40	123	0
41	Jack's New England C	New England Seafood	seafood	12 - 12 oz cans	9.65	85	0
42	Singaporean Hokkien	Leka Trading	corn products	32 - 1 kg pkgs.	14.00	26	0

REFORMAT Update

Dilwyn Jones writes:
Version 1.01 of the ReFormat rapid disk reformatter is now available.
This includes the use of a DMEDIUM_RDONLY test on suitably equipped systems (i.e. SMSQ/E) to enable a more meaningful error message when failing to reformat a write protected disk.
Sadly, this does not include QPC2, since Windows systems do not detect write protected floppies in a way in which the DMEDIUM_RDONLY extension in SBASIC can work with, so on Windows-based systems the program operates exactly the same as the previous version did.
www.dilwyn.uk6.net/files/index.html

SVSCR Suite

George Gwilt has now released version 4 of this suite.
This now supports both _psa and _pic files and has improved onscreen information. The suite can be downloaded from:
www.jms1.supanet.com

TURBO

George Gwilt has also updated the Turbo compiler.
This now allows Config Files to be updated at configuration time and not loaded separately.
http://www.jms1.supanet.com/SQLUG/gwilt/gwiltturbo.htm

Dougie Burroughs

QL-Today is sorry to learn from ***Simon N Goodwin*** of the death of Dougie Burroughs, founder and chairman of the West Midlands QL user group.
Simon writes:
Dougie was no QL expert but he was one of the most regular attendees at the club, and always willing to put in effort, both at the meetings and between them, to ensure the smooth running of the group - giving speeches, producing the annual accounts (running the gauntlet of the pedants and barrack-room lawyers in the club) and sharing in the organisation with his good friend Mike Bedford-White.
He adds that his performance will be difficult to match.

West Midlands News

Simon brought us up to date with news of the group:
As you may be aware, the West Midlands QL and 68000 user group has been meeting for more than 22 years in Birmingham, UK, and has a subscription to QL Today so that members - who may or may not also be in Quanta - are aware of what's going on in the wider QL scene (and on other Sinclairs, if Phoebus gets his way! :-). We voted to continue this support - one of our main annual expenses - at our general meeting earlier in the year, and I was mandated to send messages to QL Today and Quanta so that former members of the group might be made aware of the sad death of Dougie Burroughs, long-time QL UK (WM) chairman, earlier in 2006.

The group will continue to meet regularly twice a month, except when the first or third Monday of the month falls on a Bank Holiday, from 8 pm onwards at the Queen's Head pub on Steelhouse Lane, in the centre of Birmingham: easy to reach by car, bus (the pub is between the main bus-stops at Colmore Row and on Corporation Street) or by train; Snow Hill Station is the closest, but both New Street and Moor Street are just a few minutes walk away. QL hardware and back issues of Quanta and QL Today magazines are available at the meetings by arrangement, beer and soft drinks are of course on tap, and each year there are two special meetings when food is provided, including a party (when the club pays those who attend a pound, rather than vice versa) and the Annual General Meeting; sadly this year's AGM was the first in more than a decade not presided over by Dougie.

Mike Bedford-White has taken on responsibility for the group's finances, looking good to last another decade on current trends, and Simon N Goodwin has taken over the role of Chairman. Dougie's efforts on behalf of the group will be hard to match, but a solid core of attendees remains. New or returning members are very welcome.

Call Mike on 0121 708 2560 for details, or send £3.50 annual subscription if you wish to be added to the monthly mailing list, to:

QL and 68000 User Group (WM), 16 Westfield Road, Acocks Green, Birmingham, B27 7TL, UK

QL TODAY Index Update

Brian Kemmett has now released an index to volume 10 of QL Today.

Dilwyn Jones reports:

A complete set of indexes for QL Today right up to the end of Volume 10 is now available on my web site. The Volume 10 index, like most of the earlier versions, was prepared by Brian Kemmett. It's available as plain text or Word DOC file on my web site, or as a zipped file containing QL Quill DOC versions. Brian has indicated that he hopes to make a PDF version available soon.

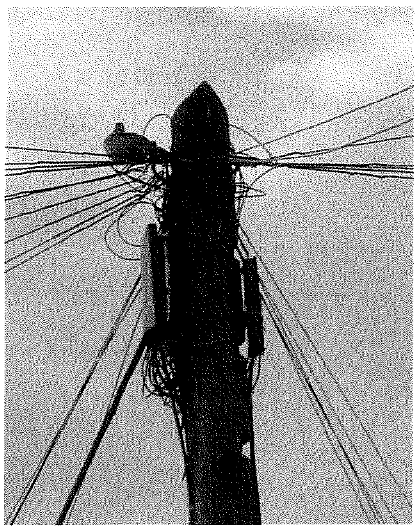
www.dilwyn.uk6.net/gen/qltoday/qltoday.html

The indexes to volumes 1 to 9 are also available on the QL Today insider documentation CD in the folder "QLTODAY". The QL Today team are grateful for Brian's hard work in indexing all 10 volumes of QL Today.

On The Move

QL Today's editor has moved. The new contact details can be found in the inside front cover of this magazine.

Following the move the editor was without an internet connection for 10 days while waiting for a telephone land line to be installed at the new premises. Installation took almost 3 hours, partly because of a tropical style rainstorm which made it unsafe for the telephone engineer to work.



In our picture you can see the telegraph pole outside the QL Today editorial office. We thought of running a "spot the editor's telephone line" competition, but we are not sure which one it is ourselves. Also in the picture is the (not so) little birdie who, we hope, will tell us all the QL news.

Quanta Blunder

It may sound like a variation of the old light bulb joke, but do you know how many Quanta members it takes to call a Special General Meeting?

Is it

(a) 100

(b) less than 15

If you do not know the correct answer then you are in good company. Neither do the Quanta committee. A QL-Today investigation has revealed that earlier this year Quanta sent an inaccurate update of the constitution to all its members.

The number of Quanta members required to call a SGM has often been a contentious issue. Originally just 25 signatures were required, but in February 1992 some members successfully called for a SGM with an agenda that many regarded as frivolous. Two months later an amendment to the constitution was approved increasing the number of signatures to 100 together with a deposit of £300. 100 signatures represented about 5% of the then membership.

By 1998 membership had declined and 100 signatures then represented about 11% of mem-

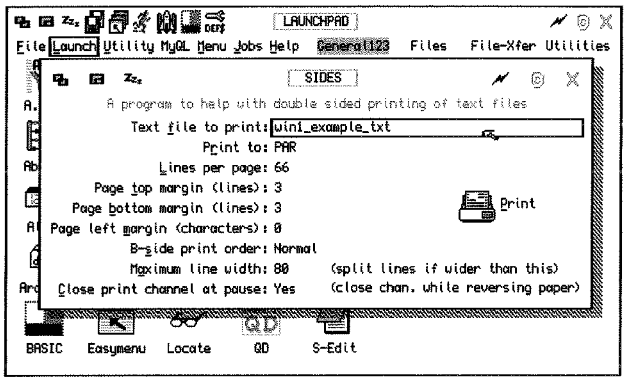
bership. An amendment to the constitution was proposed at the AGM held in Selston on 25th April 1998 reducing the required number of signatures to 5% of membership and also introducing some flexibility in the financial requirements. This motion was proposed by John Mason and seconded by Graham Underwood.
QL-Today has been unable to obtain a copy of the minutes of the 1998 AGM, but the reports of the meeting in both QL-Today and the Quanta Magazine - the latter written by the then chairman, Robin Barker - indicate that the amendment was passed.
Confusion arose last year when it was discovered that two versions of the constitution were in circulation both purporting to be the up to date constitution. This was the subject of an email correspondence between the chairman, John Mason, and committee member Geoff Wicks, now the editor of QL-Today. It is now confirmed that article 9.1 of the constitution as it appears in the constitution dated 17th April 2005 and sent to members in March 2006 is inaccurate. 5% of Quanta members and not 100 are required to call a SGM.

QL Today Advert

Quanta also made a blunder in printing an old version of the QL Today advert in the April/May Quanta Magazine. Although an up to date advert had been sent to Quanta and acknowledged by them they published the previous issue's advert. This was particularly painful for QL Today as it occurred when readers needed to renew their subscriptions.
Quanta has now made a number of changes to avoid a similar fault in the future.

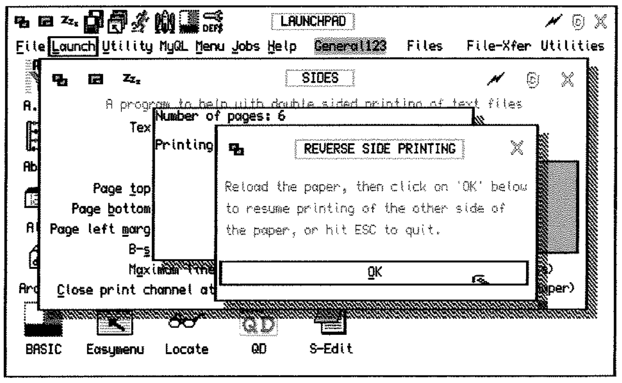
New Print Utility

Dilwyn Jones writes:
SIDES is a new program from Dilwyn Jones allowing double sided printing. It lets you print on both sides of a paper printout by first printing the odd numbered pages of a text file or basic listing, then pausing to let you reload the paper and finally printing the even numbered pages.
Depending on how paper reloading works on your printer, the even numbered pages may be printed in normal (ascending), or in reversed (descending page number), order.
The program lets you specify top, bottom and left margin along with the number of lines per page.
For emulators such as QPC2 the program has the facility to close the printer channel during the paper reversal pause, to work around the problem that QPC2 printing does not start until the print channel is closed.
Sides is a program which needs Window Manager 2 and uses System Palette colours, so you either need a fairly recent SMSQ/E or a QDOS system with Pointer Environment version 2 or later. PE v2 is freely available from Wolfgang Lenerz's SMSQ/E registrar website, or from my own website.
Sides is a Freeware program, available to download from
www.dilwyn.uk6.net/printing/index.html


LAUNCHPAD
File Launch Utility MyQL Menu Jobs Help General123 Files File-Xfer Utilities
SIDES
Print
BASIC
Easymenu
Locate
QD
S-Edit

```
A program to help with double sided printing of text files
          Text file to print: win1_example_txt
                   Print to: PAR
             Lines per page: 66
     Page top margin (lines): 3
  Page bottom margin (lines): 3
Page left margin (characters): 0
        B-side print order: Normal
        Maximum line width: 80      (split lines if wider than this)
 Close print channel at pause: Yes  (close chan. while reversing paper)
```

Figure 1 - The Sides program menu


LAUNCHPAD
File Launch Utility MyQL Menu Jobs Help General123 Files File-Xfer Utilities
SIDES
REVERSE SIDE PRINTING
OK
BASIC
Easymenu
Locate
QD
S-Edit

```
Number of pages: 6
Printing
Reload the paper, then click on 'OK' below
to resume printing of the other side of
the paper, or hit ESC to quit.
```

Figure 2 - Paper reversal and reloading pause screen)

... and Last Minute News:

PDF Index

The PDF version of ***Brian Kemmett***'s index to all volumes of QL Today has now been released. It can be downloaded from ***Dilwyn Jones***' web site:
http://www.dilwyn.uk6.net/gen/qltoday/qltoday.html
The existing separate indexes in plain text and doc file formats are still posted on the site for downloading.

Gee Graphics! (on the QL ?) - part 46

by H. L. Schaaf

Sudoku?Goduko?

I've really enjoyed George Gwilt's program, lovely graphics.
Still wasting my time trying to teach my QL to do the puzzles. Have given up on the "pure logic" approach, now willing to just make a guess every now and then, backtracking to unwind when the first guess turns out to be wrong. Time to solve is reasonable, and have not been stymied by any puzzles so far.
I'm reasonably happy with my latest effort so am 'letting it go' as sdk3c3k5_bas. Hope you enjoy the listing, and hope I can get on to other matters. I think the progress curve for the easy puzzles look a bit like Lame curves.
Just for comparison: program sdk3c3k5_bas on the easy puz80_sdk.

Time to solve	System
469 seconds	JSU Trump Card
121	JSU Gold Card
45	JSU Super Gold Card
20	HBA 2.89 Gold Card
8	HBA 2.91 Super Gold Card
4	HBA 3.09 QPC 3.30 (laptop/Win98SE, 233Mhz))

I see there are puzzles (Goduko?) that use letters instead of or in addition to the usual digits 1 through 9, so I've written the program to handle any 9 symbols (or characters). Limiting myself to the 9 x 9 grid for now. Some puzzles don't have all 9 different starting values, so we've allowed for adding extra symbols when needed.

Try solving this easy one:

```
. . Q   . . o   . L a
. . .   . L .   . . .
. . .   . . a   . o T

o T .   . y !   Q . .
. . .   . . .   . . .
! . .   . a .   . T .

. . .   o . _   . a y
a o .   d . .   ! . _
. Q .   . . .   . . L
```

I'll bet readers, especially Geoff Wicks, can come up with a lot of 9 character expressions, all 9 characters unique. What say? Please keep them polite! (but need not be politically correct).

I tried the templates of Norman Dunbar's program from the last issue of QL Today with HBA 2.89 & Gold Card and got the following results (time in seconds, guesses made):

Level--›	Easy	Medium	Hard	Difficult
Template				
1	24, 0	32, 1	48, 1	52, 3
2	22, 0	28, 1	28, 2	29, 1
3	19, 0	36, 1	39, 2	41, 2

Symmetries do matter, so rotations, reflections, etc. will cause the puzzle solving time to vary.

Scientific American June 2006 pages 80-87 "The Science behind Sudoku" by Jean-Paul Delahaye, mentions a subset concept. I tried it but it doesn't eliminate guessing, and usually (not always) takes more time.

Broken Link - Ephemeral website
Hirofumi Fujiwara
(www.pro.or.jp/~fuji/sudoku/index-eng.html)
showed how to do it by hand, with reverence for beauty and symmetry. I wish I'd saved instead of just bookmarking, now it is a broken link on the web, due to the death of the author.

I use a descriptive 'header' line before the puzzle, and all the filenames are of the form "puz###_sdk" where ### are numbers.
This is what my text file for the puzzle above looks like:

```
80 puz80_sdk  Goduko for QL Today Easy
..Q..o.La
....L....
.....a.oT
oT..y!Q..
.........
!...a..T.
...o._.ay
ao.d..!._
.Q......L
```

In the PROCedure get_file, make adjustments to suit your own preferences.

Listing sdk3c3k5_bas

```
100 REMark sdk3c3k5_bas
110 REMark H. L. Schaaf
120 REMark July 12, 2006
130 REMark for GG#46 QL Today
140 :
150 PAPER #0,0 : INK #0, 4: MODE 4: WMON
160 CSIZE 0,0 : PAPER 0 : INK 7 : CLS
170 CSIZE #2;0,0 : PAPER#2,0 : INK#2,7
180 SCALE #2,100,0,0 : CLS#2
190 :
200 DIM puz$(9,9) :REMark the 'givens'
210 DIM sdk$(9,9) :REMark the working grid
220 DIM sol$(9,9) :REMark a solution
230 DIM row$(9,9) :REMark from file
240 DIM guess$(80,8) :REMark is 80 enough?
250 REMark use 7 to 8 for guess move number 1
    to 81 ?
260 REMark use 1,2 for row, column
270 REMark use 3 for number of times tried
280 REMark use 4,5,6 for try 1st or 2nd (or
    3rd ?)
290 REMark fork = guess point
300 fork = 0 : doubs = 0 : trips = 0
310 DIM move$(81,10) : REMark keep track of
    events, could use 14 fewer ?
320 REMark keep trial and error record of
    false paths tried in guesses$
330 guesses$=''
340 REMark record of logical options at each
    sweep
350 logop$=''
360 move_num = 0  : REMark move number
370 guessmove = 1
380 guess_num = 0 : REMark guesses made
390 oldcellsempty = 81
400 sweeps = 0
410 :
420 INPUT #0;'Number of the puzzle ? '; puzn$
430 f$ = 'puz'&puzn$&'_sdk'
440 PRINT #0;f$
450 get_file
460 getpuzchrs
470 :
480 start = DATE
490 inorni
500 inorni : show_start : lookit : CLS
510 show_opts : log_opts : logop$=''
520 :
530 REPeat sweep
540 show_logop
550 logop$=logop$&'#'&logopt%
560  IF NOT(cellsempty) : EXIT sweep
570  sing : REMark get any singles
580  cellchange = oldcellsempty - cellsempty
590  IF NOT(cellchange) THEN
600    Doub
610  END IF
620  oldcellsempty = cellsempty
630  lookit
640  show_opts
650  sweeps = sweeps +1
660 END REPeat sweep
670 :
680 timed = DATE-start
690 CLS#0: PRINT #0; numsrcdate$
700 PRINT #0; timed & ' seconds ' & sweeps &'
    sweeps'
710 PRINT#0; LEN(guesses$)DIV 10;' false moves
    ';guess_num;' guesses   ';fork;' forks  '
720 PRINT #0;'any key for moves ':PAUSE
730 CLS :
740 FOR i = 0 TO (move_num+1) STEP 3
750  FOR j = 1 TO 3
760   PRINT move$(i+j)!!!!
770  END FOR j
780  PRINT
790 END FOR i
800 AT#2, 11, 0 : CLS#2,3 :CLS#2,2
810 AT#2, 18, 4 : PRINT#2; 'any key for
    progress chart ' : PAUSE
820 show_prog
830 :
840 REMark Singles were a logical choice,
850 REMark Doubles & Triples were guesses
860 DIM  fig$(3,7)
870 fig$(1)='Singles'
880 fig$(2)='Doubles'
890 fig$(3)='Triples'
900 DIM fig(3) : REMark Single, Double, Triple
910 FOR i = 1 TO move_num
920  IF 'S' INSTR move$(i,9 TO 10):fig(1)=
     fig(1)+1
930  IF 'D' INSTR move$(i,9 TO 10):fig(2)=
     fig(2)+1
940  IF 'T' INSTR move$(i,9 TO 10):fig(3)=
     fig(3)+1
950 END FOR i
960 FOR i = 1 TO 3
970  PRINT #0;fig(i)!!fig$(i),
980 END FOR i
990 :
1000 REMark save the solution
1010 FOR i = 1 TO 9
1020  sol$(i)=sdk$(i)
1030 END FOR i
1040 :
1050 PRINT\ ,"puzzle",,"solution"\\
1060 FOR i = 1 TO 9
1070  PRINT, puz$(i),sol$(i)
1080 END FOR i
1090 :
1100 REMark could add menu to print, save
1110 REMark moves, solutions, etc.
1120 REMark leave as exercise for readers?
1130 STOP
1140 :
1150 REMark logical options at each sweep
1160 DEFine PROCedure log_opts
1170  LOCal i,j
1180  DIM logop%(9)
1190  logopt% = 0
1200  FOR i = 1 TO 9
1210   FOR j = 1 TO 9
1220    IF sdk$(i,j)=mfg$ THEN
1230     lop% = LEN(opt$(i,j))
1240     logopt% = logopt% + lop%
1250     logop%(lop%)=logop%(lop%)+1
1260    END IF
1270   END FOR j
1280  END FOR i
1290 END DEFine log_opts
```

```
1300 :
1310 DEFine PROCedure show_opts
1320  REMark what options logically exist for
      each cell ?
1330  LOCal i,j
1340  CLS
1350  FOR i = 1 TO 9
1360   FOR j = 1 TO 9
1370    AT 2*i,(4*j)-4
1380    IF LEN(opt$(i,j)) THEN
1390     IF LEN(opt$(i,j))= 1 THEN
1400      INK 4
1410     ELSE
1420      INK 7
1430     END IF
1440     PRINT opt$(i,j);
1450    ELSE
1460     INK 7
1470     PRINT '.';
1480    END IF
1490   END FOR j
1500  END FOR i
1510 END DEFine show_opts
1520 :
1530 DEFine PROCedure set_grid
1540  LOCal i,j
1550  FOR i = 1 TO 9
1560   FOR j = 1 TO 9
1570    sdk$(i,j) = row$(i,j)
1580    puz$(i,j) = sdk$(i,j)
1590    PRINT sdk$(i,j);
1600   END FOR j
1610   PRINT
1620  END FOR i
1630 END DEFine set_grid
1640 :
1650 DEFine PROCedure show_start
1660  LOCal i,j
1670  REMark basic white ink on black paper
1680  PAPER #2,0 : INK #2,7 : CLS#2
1690  REMark black ink, on white paper to show
      'givens'
1700  PAPER #2, 7 : INK #2, 0
1710  FOR i= 1 TO 9
1720   FOR j = 1 TO 9
1730    IF puz$(i,j)INSTR gly$ THEN
1740     AT#2; i,2*j+1: PRINT#2; puz$(i,j)
1750    END IF
1760   END FOR j
1770  END FOR i
1780  REMark back to white ink on black paper
1790  PAPER #2, 0 : INK #2,7
1800 END DEFine show_start
1810 :
1820 DEFine FuNction box(row,col)
1830  REMark given a cell's row and column,
1840  REMark returns the box number the cell
      is in.
1850  RETurn  (3*((row-1) DIV 3)) +((col-1)
      DIV 3) + 1
1860 END DEFine : REMark box(row,col)
1870 :
1880 DEFine FuNction topleft(box_num)
1890  REMark given a box number,
1900  REMark returns top left row and column
1910  tlr = 4+(((box_num-1)DIV 3)-1)*3
1920  tlc = 1+((box_num-1)MOD 3)*3
1930  RETurn tlr
1940  RETurn tlc
1950 END DEFine :REMark topleft(box_num)
1960 :
1970 DEFine FuNction cellsempty
1980  LOCal i,j
1990  count = 0
2000  FOR i = 1 TO 9
2010   FOR j = 1 TO 9
2020    IF (sdk$(i,j)=mfg$) : count = count +1
2030   END FOR j
2040  END FOR i
2050  RETurn count
2060 END DEFine :REMark cellsempty
2070 :
2080 DEFine PROCedure sing
2090  LOCal i,j
2100  FOR i = 1 TO 9
2110   FOR j = 1 TO 9
2120    IF (LEN(opt$(i,j)) = 1) THEN
2130     IF (sdk$(i,j) = mfg$) THEN
2140      w$='S'
2150      place i,j,(opt$(i,j)),4
2160     END IF
2170    END IF
2180   END FOR j
2190  END FOR i
2200 END DEFine sing
2210 :
2220 REMark when no more singles,
2230 REMark guess one of a Double, pick one
     for tryout
2240 REMark if it works, lucky , otherwise it
     must be the other
2250 DEFine PROCedure Doub
2260  LOCal i, j
2270  fork = fork + 1
2280  tripped = 0
2290  found2 = 0
2300  choice3$ = ''
2310  FOR i = 1 TO 9
2320   FOR j = 1 TO 9
2330    IF LEN(opt$(i,j))= 2
2340     IF (sdk$(i,j)=mfg$) THEN
2350      guess_num = guess_num + 1
2360      found2 = 1
2370      row =i : col =j
2380      choice1$ = opt$(i,j,1)
2390      choice2$ = opt$(i,j,2)
2400      AT 2*i,(4*j)-4 : INK 2 :PRINT
          opt$(i,j):INK 7
2410      EXIT i
2420     END IF
2430    END IF
2440   END FOR j
2450  END FOR i
2460 REMark if no doubles as option ?
2470 REMark try best? of 3
2480 REMark IF NOT(found2) : unwind :REMark
     trip
2490 IF NOT(found2) : trip
2500 REMark pick one and save the others for
     later
2510 IF NOT(tripped) THEN
2520  w$='D'&(0+guess_num)&choice1$
2530 ELSE
2540  w$='T'&(0+guess_num)&choice1$
2550 END IF
2560 guessmove = move_num+1
2570 guess$(fork,7 TO 8)=guessmove :
     guess$(fork,1)=row
```

```
2580 guess$(fork,2)=col
2590 guess$(fork,3) = '1' : REMark first use
     of guess
2600 guess$(fork,4)=choice1$
2610 guess$(fork,5)=choice2$
2620 IF tripped : guess$(fork,6)=choice3$
2630 g_row = row : g_col = col
2640 IF NOT (tripped) THEN
2650  place row,col,choice1$,2
2660 ELSE
2670  place row,col,choice3$,7
2680 END IF
2690 END DEFine Doub
2700 :
2710 REMark triples when no doubles ?
2720 DEFine PROCedure trip
2730  LOCal i,j
2740  found3 = 0
2750  fork = fork + 1
2760  trips = trips +1
2770  tripped = 1
2780  FOR i = 1 TO 9
2790   FOR j = 1 TO 9
2800    IF LEN(opt$(i,j))= 3 THEN
2810     IF (sdk$(i,j)=mfg$) THEN
2820      guess_num = guess_num + 1
2830      trip_num = guess_num
2840      row =i : col =j
2850      choice1$ = opt$(i,j,1)
2860      choice2$ = opt$(i,j,2)
2870      choice3$ = opt$(i,j,3)
2880      found3 = 1
2890      EXIT i
2900     END IF
2910    END IF
2920   END FOR j
2930  END FOR i
2940 IF NOT(found3): PRINT #0;'no triples ?' :
     PAUSE 300 : unwind
2950 END DEFine trip
2960 :
2970 :
2980 DEFine PROCedure place(row,col,plc$,nk)
2990  LOCal i , j, ii, jj
3000  REMark check for conflicts first ?
3010  conflict = 0
3020  IF plc$=mfg$ : conflict = conflict
      +1:PAUSE
3030  :
3040  REMark check in box ?
3050  box_num =  box(row,col)
3060  tlr = topleft(box_num)
3070  FOR ii = tlr TO tlr + 2
3080   FOR jj = tlc TO tlc + 2
3090    IF sdk$(ii,jj) = plc$ THEN
3100     conflict = conflict + 2
3110     EXIT ii
3120    END IF
3130   END FOR jj
3140  END FOR ii
3150  :
3160  IF NOT(conflict) THEN
3170   REMark check row
3180   FOR j = 1 TO 9
3190    IF sdk$(row,j)=plc$ THEN
3200     conflict = conflict + 4
3210     EXIT j
3220    END IF
3230   END FOR j
3240  END IF
3250  :
3260  IF NOT(conflict)
3270   REMark check column
3280   FOR i = 1 TO 9
3290    IF sdk$(i,col)= plc$ THEN
3300     conflict = conflict + 8
3310     EXIT i
3320    END IF
3330   END FOR i
3340  END IF
3350  :
3360  IF conflict AND fork THEN
3370 REMark  PRINT #0;'conflict AND fork
     ';conflict
3380   unwind
3390  ELSE
3400   IF NOT(conflict)
3410    subs_str plc$,row$(row,col)
3420    sdk$(row,col) = plc$
3430    join_str plc$, inro$(row)
3440    oust_str plc$, niro$(row)
3450    join_str plc$, inco$(col)
3460    oust_str plc$, nico$(col)
3470    join_str plc$, inbo$(box(row,col))
3480    oust_str plc$, nibo$(box(row,col))
3490    opt$(row,col) = ''
3500    move_num = move_num + 1
3510    move$(move_num)=move_num&'r'&row&'c'
        &col&'='&plc$&' '&w$
3520    INK#2; nk: AT#2; row,2*col+1 :PRINT#2;
        plc$ :INK#2; 7
3530    AT 2*row,(4*col)-4 :PRINT'.   ';
3540   END IF
3550  END IF
3560 END DEFine place
3570 :
3580 DEFine PROCedure unwind
3590  LOCal i, j, k
3600  REMark which is earliest unchanged fork?
3610  REMark collect all the moves since the
      guess
3620  REMark redo just the most recent
      unchanged
3630  FOR i = move_num TO 1 STEP -1
3640   IF guess$(i,3)='1' THEN
3650    guessmove = guess$(i,7 TO 8)
3660    REMark get row col, choice2$ for use
        below:
3670    g_row = guess$(i,1) : g_col = guess$(i,2)
3680    choice1$ = guess$(i,4) :choice2$ =
        guess$(i,5)
3690    guess$(i,3)= '2'
3700    fork = i
3710    FOR j = i + 1 TO move_num
3720     FOR k = 1 TO 7
3730      guess$(j,k) = ''
3740     END FOR k
3750    END FOR j
3760    EXIT i
3770   END IF
3780  END FOR i
3790  REMark keep record of false moves ?
3800  FOR i = guessmove TO move_num
3810   guesses$=guesses$&move$(i)&' '
3820  END FOR i
3830  REMark go back to where choice was made
      and change
3840  FOR i = move_num TO guessmove  STEP -1
```

```
3850  m$=move$(i)
3860  row = m$(('r'INSTR m$)+1)
3870  col = m$(('c'INSTR m$)+1)
3880  REMark erase in #2
3890  AT#2; row,2*col+1 :PRINT#2;' '
3900  sdk$(row,col)=mfg$
3910  subs_str mfg$, row$(row,col)
3920  join_str m$(('='INSTR m$)+1),opt$(row,col)
3930  move$(i)=''
3940 END FOR i
3950 move_num = guessmove - 1
3960 guess_num=guess_num+1
3970 w$ = 'd'&(0+guess_num)&choice2$
3980 place g_row,g_col,choice2$,242
3990 sdk$(g_row,g_col)=choice2$
4000 inorni
4010 lookit
4020 show_opts
4030 REMark -- inelegant !! ?? must be a
     better way to program this ?
4040 GO TO 530
4050 END DEFine unwind
4060 :
4070 DEFine PROCedure get_file
4080 LOCal i,j
4090 REMark read in text file of puzzle
4100 REMark with header and/or footer ?
4110 f_n = FOP_IN ( f$ )
4120 IF f_n > 0 THEN
4130  PRINT #0;' opening #';f_n,
4140 ELSE
4150  PRINT #0;' Error ';f_n;' opening file'
4160 END IF
4170 REMark following gets 'header'
4180 INPUT #f_n,numsrcdate$
4190 PRINT #0;numsrcdate$
4200 REMark following gets puzzle
4210 FOR i = 1 TO 9
4220  INPUT #f_n ; puz$(i)
4230  sdk$(i) = puz$(i)
4240  row$(i) = puz$(i)
4250 END FOR i
4260 CLOSE #f_n
4270 END DEFine get_file
4280 :
4290 REMark what are the symbols, glyphs, etc.
     used ?
4300 DEFine PROCedure getpuzchrs
4310 LOCal i,j
4320 REMark Goduko characters (glyphs)
4330 REMark can be any symbols A,1,#,etc.
4340  gly$=row$(1)
4350  FOR i = 2 TO 9
4360   join_str  row$(i),gly$
4370  END FOR i
4380  long$ = gly$
4390  REMark which is the null glyph?
4400  REMark assume is most frequent glyph
      (mfg$) ?
4410  uniq_str gly$
4420  DIM cc%(LEN(gly$))
4430  FOR i = 1 TO LEN(long$)
4440   FOR j = 1 TO LEN(gly$)
4450    IF long$(i)=gly$(j) THEN
4460     cc%(j)=cc%(j)+1
4470     EXIT j
4480    END IF
4490   END FOR j
4500  END FOR i
4510  REMark what is most common ?
4520  maxi = 0
4530  FOR i = 1 TO DIMN (cc%)
4540   IF cc%(i)>maxi THEN
4550    maxi = cc%(i)
4560    cc%(0)= i
4570   END IF
4580  END FOR i
4590  mfg$ = gly$(cc%(0))
4600  oust_str mfg$,gly$
4610  check_gly
4620 END DEFine getpuzchrs
4630 :
4640 DEFine PROCedure inorni
4650 LOCal i, j
4660 REMark in or not in (ni) row, col, box
4670 DIM inro$(9,9)
4680 DIM niro$(9,9)
4690 DIM inco$(9,9)
4700 DIM nico$(9,9)
4710 DIM inbo$(9,9)
4720 DIM nibo$(9,9)
4730 FOR i = 1 TO 9
4740  REMark put them all in
4750  niro$(i)=gly$
4760  nico$(i)=gly$
4770  nibo$(i)=gly$
4780  FOR j = 1 TO 9
4790   sym$ = row$(i,j)
4800   IF sym$<>mfg$ THEN
4810    join_str sym$, inro$(i)
4820    join_str sym$, inco$(j)
4830    join_str sym$, inbo$(box(i,j))
4840   END IF
4850  END FOR j
4860 END FOR i
4870 FOR i = 1 TO 9
4880  REMark now deselect them
4890  oust_str inro$(i),niro$(i)
4900  oust_str inco$(i),nico$(i)
4910  oust_str inbo$(i),nibo$(i)
4920 END FOR i
4930 END DEFine inorni
4940 :
4950 REMark a general procedure to take
4960 REMark specific character(s) out of
4970 REMark a given string
4980 DEFine PROCedure oust_str (oust$ , from$)
4990 LOCal len_ou,len_fr, n, i
5000 len_ou = LEN(oust$)
5010 len_fr = LEN(from$)
5020 IF len_fr AND len_ou THEN
5030  n = oust$ INSTR from$
5040  FOR i = 1 TO len_ou
5050   len_fr = LEN(from$)
5060   n = oust$(i) INSTR from$
5070   IF n THEN
5080    IF ((n>1) AND (n<len_fr)) THEN
5090     from$ = from$(1 TO (n-1))&from$
         ((n+1) TO)
5100    END IF
5110    IF (n = 1) : from$ = from$(2 TO)
5120    IF (n = len_fr) : from$ = from$(1 TO
        n-1)
5130   END IF
5140  END FOR i
5150 END IF
5160 END DEFine oust_str
5170 :
```

JOCHEN MERZ SOFTWARE

Kaiser-Wilh.-Str. 302 D-47169 Duisburg
Tel. 0203 502011 Fax 0203 502012
http://smsq.j-m-s.com smsq@j-m-s.com

"Online Grabbelkiste": I am unpacking boxes which have not been touched for about two years - you may remember, J-M-S moved in September/October 2004.

I have found many interesting items in larger quantities, e.g. ZX8301, ZX8302 ... unused!
Boxes of DD disks, unused, of course. Various brands, various colours.
Internal IDE ZIP drive and ZIP disks.
Multi-IO card with ISA slot and jumpers, so that it can be used in the Q40 or Q60 to support four serial ports.

If any of the above items sound interesting to you, please contact me via email.
Let me know how many items you want, and we discuss price and shipping.

SMSQ/E

QPC2 Version 3 with SMSQ/E ... now only **EUR 59,90**
QPC2 Version 3 upgrade from Version 2 ... only **EUR 19,90**
QPC2 Version 3 upgrade from Version 1 ... only **EUR 39,90**
QPCPrint ... still only **EUR 39,90**

Please note that QPCPrint will be shipped on CD only.
QPC2 will also be shipped on CD by default now, as more and more systems (especially notebooks) do not have floppy disks drives built in anymore. If you prefer QPC2 on HD floppy disks, please state with your order. Free updates are available on Marcel's Website www.KILGUS.net If you prefer updates on CD or floppy disk, send in your master medium together with 4 international reply coupons to cover return postage, medium and packaging.

SMSQ/E for ATARI or GoldCard or SuperGoldCard or QXL ...
now only **EUR 39,90**

Colour drivers now included as standard for (Super)GoldCard versions
For shipping inside Europe, EUR 3,- will be added for postage and packaging.
For shipping outside Europe, EUR 6,- will be added for p & p.

If you wish to place your order via internet, please do not send your credit card details in an email!
Please use the secure contact form on the SMSQ homepage: SMSQ.J-M-S.COM
We accept VISA, MasterCard and Diners Club!

```
5180 REMark substitution in string
5190 DEFine PROCedure subs_str (new$ ,old$)
5200  old$ = new$
5210 END DEFine subs_str
5220 :
5230 REMark join string$ ?
5240 DEFine PROCedure join_str (join$, target$)
5250  target$ = target$ & join$
5260 END DEFine join_str
5270 :
5280 REMark remove duplicates from string
5290 DEFine PROCedure uniq_str (dup$)
5300  LOCal i,u$,chr
5310  u$ = ''
5320  REMark only 255 possibilities ?
5330  DIM chr%(255)
5340  FOR i = 1 TO LEN(dup$)
5350   chr = CODE(dup$(i))
5360   chr%(chr)=chr%(chr)+1
5370  END FOR i
5380  REMark now pick them off in the same
      order ?
5390  FOR i = 1 TO LEN (dup$)
5400   chr = CODE(dup$(i))
5410   IF chr%(chr) THEN
5420    u$ = u$ & dup$(i)
5430    chr%(chr)= 0
5440   END IF
5450  END FOR i
5460  dup$ = u$
5470 END DEFine uniq_str
5480 :
5490 REMark gather the options into opt$
5500 DEFine PROCedure get_options
5510  LOCal i,j,k
5520  DIM opt$(9,9,9)
5530  FOR i = 1 TO 9
5540   FOR j = 1 TO 9
5550    IF sdk$(i,j)<>mfg$ THEN
5560     opt$(i,j)=''
5570    ELSE
5580     comm3 niro$(i),nico$(j),nibo$(box
         (i,j)),opt$(i,j)
5590    END IF
5600   END FOR j
5610  END FOR i
5620 END DEFine get_options
5630 :
5640 REMark find what is common to all 3
     string$ = row, & col, & box
5650 DEFine PROCedure comm3 (str1$,str2$,
     str3$,inall3$)
5660  LOCal i,j,k,work$,comm$
5670  work$ = ''
5680  join_str str1$,work$
5690  join_str str2$,work$
5700  join_str str3$,work$
5710  comm$ =  work$
5720  uniq_str work$
5730  DIM com%(LEN (work$))
5740  FOR i = 1 TO LEN(comm$)
5750   k = (comm$(i) INSTR work$)
5760   IF k THEN
5770    com%(k) = com%(k)+1
5780   END IF
5790  END FOR i
5800 REMark it takes 3 to be common to all
5810  inall3$ = ''
5820  FOR i = 1 TO DIMN(com%)
5830   IF com%(i)=3 THEN
5840    join_str work$(i),inall3$
5850   END IF
5860  END FOR i
5870 END DEFine comm3
5880 :
5890 REMark only one of a kind ?
5900 REMark taking 9 cells at a time ?
5910 DEFine PROCedure only1per (rcb$)
5920 LOCal i,j,ii,jj,k,kk,work$,per
5930 per = rcb$ INSTR 'rcb'
5940 :
5950 FOR i = 1 TO 9
5960  work$=''
5970  IF per = 1 THEN
5980   REMark by rows
5990   FOR j =1 TO 9
6000    join_str opt$(i,j),work$
6010   END FOR j
6020   only$=work$
6030  END IF
6040 :
6050  IF per = 2 THEN
6060   REMark by columns
6070   FOR j =1 TO 9
6080    join_str opt$(j,i),work$
6090   END FOR j
6100   only$=work$
6110  END IF
6120 :
6130  IF per = 3 THEN
6140   REMark by boxes
6150   FOR ii = topleft(i) TO tlr+2
6160    FOR jj = tlc TO tlc+2
6170     join_str opt$(ii,jj),work$
6180    END FOR jj
6190   END FOR ii
6200   only$=work$
6210  END IF
6220 :
6230  IF LEN(only$) THEN
6240   REMark there is a cell to change
6250   uniq_str work$
6260   REMark make array and count
6270   lone$ = ''
6280   DIM one%(LEN(work$))
6290   FOR j = 1 TO LEN(only$)
6300    k = only$(j) INSTR work$
6310    IF k THEN
6320     one%(k)=one%(k)+1
6330    END IF
6340   END FOR j
6350   FOR j = 1 TO DIMN(one%)
6360    IF one%(j) = 1 THEN
6370     join_str  work$(j),lone$
6380    END IF
6390   END FOR j
6400  END IF
6410 :
6420  k =  LEN(lone$)
6430  IF k THEN
6440   REMark locate and change opt$
6450   FOR j = 1 TO k
6460 :
6470    IF per = 1 THEN
6480     FOR jj = 1 TO 9
6490      IF lone$(j) INSTR opt$(i,jj) THEN
6500       opt$(i,jj)=lone$(j)
6510       EXIT jj
```

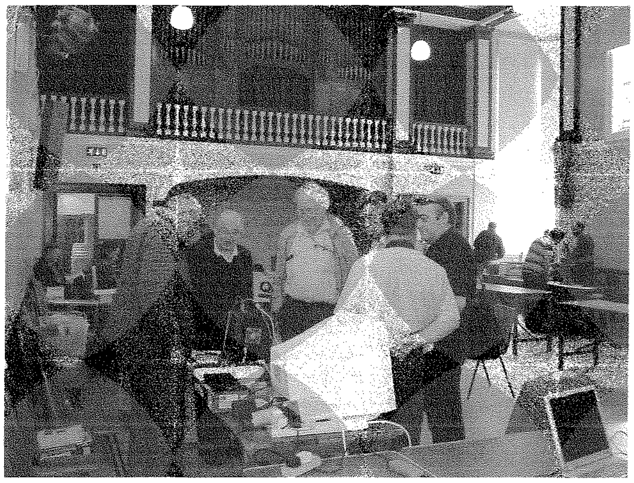
```
6520      END IF
6530     END FOR jj
6540    END IF
6550 :
6560    IF per = 2 THEN
6570     FOR jj = 1 TO 9
6580      IF lone$(j) INSTR opt$(jj,i) THEN
6590       opt$(jj,i)=lone$(j)
6600       EXIT jj
6610      END IF
6620     END FOR jj
6630    END IF
6640 :
6650    IF per = 3 THEN
6660     FOR jj = topleft(i) TO tlr+2
6670      FOR kk  = tlc TO tlc+2
6680       IF lone$(j) INSTR opt$(jj,kk) THEN
6690        opt$(jj,kk)=lone$(j)
6700        EXIT kk
6710       END IF
6720      END FOR kk
6730     END FOR jj
6740    END IF
6750   END FOR j
6760  END IF
6770 END FOR i
6780 END DEFine only1per
6790 :
6800 REMark look it over ?
6810 DEFine PROCedure lookit
6820  LOCal i
6830  get_options
6840  per$ = 'rcb'
6850  FOR i = 1 TO 3
6860   only1per per$(i)
6870  END FOR i
6880 END DEFine lookit
6890 :
6900 REMark check_gly
6910 DEFine PROCedure check_gly
6920  REMark if not exactly 9 symbols ?
6930  add_char = 57
6940  REPeat gly_check
6950   IF (LEN(gly$)) > 9 : PRINT #0;'too many
       symbols ?':STOP
6960   IF (LEN(gly$))= 9 : EXIT gly_check
6970   REPeat try_another
6980    IF CHR$(add_char) INSTR gly$ THEN
6990     add_char = add_char - 1
7000    ELSE
7010     EXIT try_another
7020    END IF
7030   END REPeat try_another
7040   gly$=gly$&(CHR$(add_char))
7050  END REPeat gly_check
7060 END DEFine check_gly
7070 :
7080 REMark show logical options
7090 DEFine PROCedure show_logop
7100  LOCal i
7110  log_opts
7120  FOR i = 11 TO 13
7130   AT #2; i,0  :CLS#2, 3
7140  END FOR i
7150  AT #2; 11,2
7160  PRINT#2; logopt%;' options'
7170  FOR i = 1 TO 9
7180   AT #2; 12,((i-1)*4)+2
7190   PRINT #2; i
7200   AT #2; 13,((i-1)*4)+2
7210   PRINT#2; logop%(i)
7220  END FOR i
7230  POINT#2;2,logop%(1)
7240  FOR i = 2 TO 9
7250   LINE TO #2;10*(i-1)+2,logop%(i)
7260  END FOR i
7270 END DEFine show_logop
7280 :
7290 REMark show progress of solving
7300 DEFine PROCedure show_prog
7310  LOCal i
7320  REMark build array ?
7330  dlo = 0
7340  FOR i = 1 TO LEN (logop$)
7350   IF logop$(i) = '#'
7360    dlo = dlo + 1
7370   END IF
7380  END FOR i
7390  DIM dlo%(dlo)
7400  ndxat = 1 : count = 0
7410  REPeat build_dlo
7420   count = count + 1
7430   nxoct = ndxat+1+ '#' INSTR logop$(
       (ndxat+1) TO)
7440   IF NOT(nxoct-1-ndxat) THEN
7450    dlo%(count) = logop$((ndxat+1) TO)
7460    EXIT build_dlo
7470   END IF
7480   dlo%(count) = logop$((ndxat+1) TO
       (nxoct-2))
7490   ndxat = nxoct-1
7500  END REPeat build_dlo
7510  REMark clear out old logops
7520  AT #2; 11,0 : CLS#2,2
7530  SCALE#2; 2.5*dlo%(1),0,0
7540  stepsz =2*dlo%(1)/ DIMN(dlo%)
7550  REMark show in #2
7560  INK #2,4
7570  POINT #2; 0,dlo%(1)
7580  FOR i = 2 TO DIMN(dlo%)
7590   LINE TO #2; stepsz*(i-1),dlo%(i)
7600  END FOR i
7610  CLS: PRINT, '  options per sweep'
7620  FOR i = 1 TO DIMN(dlo%)
7630   AT 2+((i-1) DIV 10),2+((cyc(i,10)-1)*4)
7640   PRINT dlo%(i)
7650  END FOR i
7660 :
7670 END DEFine show_prog
7680 :
7690 DEFine FuNction cyc (number%, cycle_length)
7700  REMark option base 1
7710  REMark PRINT number%, cycle_length
7720  RETurn ((number%-1)MOD cycle_length)+1
7730 END DEFine : REMark cyc
7740 :
7750 REMark end of listing sdk3c3k5_bas
```

10th Hove Show

by Geoff Wicks

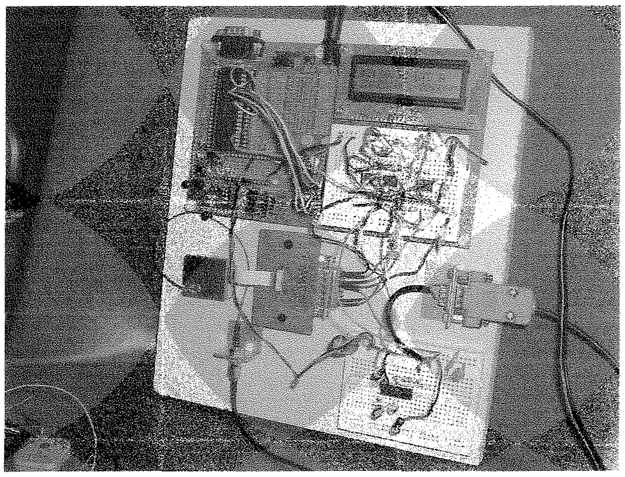
If size matters then Portslade Town Hall has a big one, and that frustrates me. I am writing about organs, of course. The frustration is that this organ seems impossible to play. There is a space for the organist to sit, but no obvious way for him to get to the keyboard.

Perhaps the answer to this problem is to ask to use it at next year's Hove show. Our two itinerant musicians, Tony "Angel Face" Firshman and Roy "Weird" Wood, could entertain us. It would be an interesting clash of styles, although I am not sure whether you would call it a concert or a gig.
Portslade Town Hall has now become an established show venue, and although the roof is not in imminent danger of collapse, Sackville Hotel style, it has a couple of deficiencies. The important one for us is the poor provision of electrical sockets. The unimportant one is that the maximum permitted capacity of the hall is 150 people. This is hardly a problem for present day QL shows. Attendance at Hove was touching 30.
Hove is the one show in the year where you can guarantee most of the traders will be present and this year was no exception. The main absentee was Jochen Merz, whose turn it was to have problems with his car. (He hopes there will be a UK show in the autumn for him to attend.) And there was a bit of luck this year for those of us who came by public transport. The usual engineering works leading to all London to Brighton trains being diverted via Penzance (just a slight exaggeration) had taken place a couple of weeks previously.
Surprise trader at the show was Bill Richardson, who came back from retirement, to sell the last of his stock. At least that was his story, because the main thing on his stall, apart from the last of his Z88's and peripherals, was masses of anti-European Union propaganda. Bill failed to convert this ardent European, but we avoided an unseemly spat by the old trick of finding something we could agree on. We went through the main British politicians one by one and agreed they were a useless bunch. But then, what can you expect when there is not a single QL-er among them?



Another welcome face at the show was Roger Godley, who is updating the Psion programs for GD2 high resolution screens. His machine gave a new meaning to "a black box QL". It was a PC running QPC2 on a flat screen all built into a black wooden box with a handle for easy transportation. There were so many wires coming out if it that it looked like a prop for a 1960's science fiction movie.
Roger had been asked to talk about his work but he preferred to do this with small groups rather than in the lecture room. There was no shortage of people wishing to talk to him.

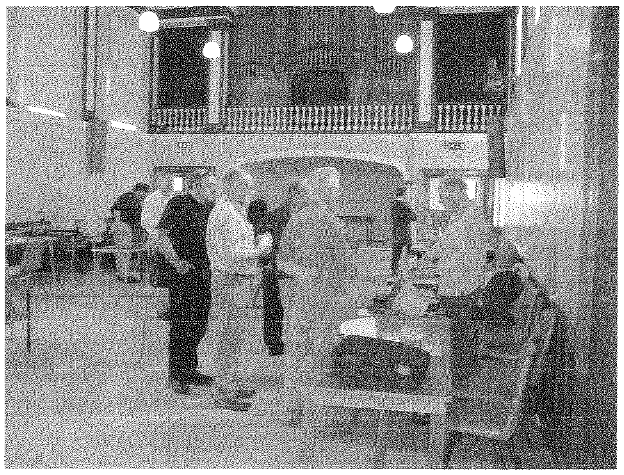
Talks and demonstrations are back in fashion at QL shows and organisers are now expected to beg, borrow or hire a computer projector. Hove was no exception. Rather too late it was discovered that Portslade Town Hall does not possess a screen and Roy Wood had to improvise with two curtains held in place by chairs. It did the job, but just a thought. Doesn't Quanta have the funds to buy a projector and screen for use at future shows?



First to make use of the projector was Hugh Rooms who has married the QL to the latest satellite technology. Hugh is an amateur radio enthusiast who has built a receiver for GPS signals. This input from this is fed into a basic program in QPC2 for analysis and display. Hugh was not sure whether he would be able give a "live" demonstration at the show, and had prepared a simulation. On the left hand of the screen was a display of the locations of the detected satellites, and on the right hand side the processed information from them.
The first attempts at a live demonstration were only partly successful. Hugh could locate some satellites and get a signal but was unable to receive the data to locate a position. After some experimentation Tony Firshman placed the aerial outside a window and within a few minutes several satellites were located and the latitude and longitude of the hall were displayed on the screen.
This was a highly technical lecture mainly because Hugh had to go into some detail over mapping techniques, which are more complicated than most of us envisage. For example two types of data have to be collected. One is the almanac which he compared to a railway timetable and the other the real time situation as happens, for example, when a train is running late. He also spent a lot of time talking about precision measurement pointing out how archaeological sites can "move" over a number of years and how the Greenwich Meridian had been in five different places during its lifetime.

The other lecture of the day was on more familiar QL territory. Roy Wood demonstrated the latest version of the QDT File Manager. QDT has come a long way since it was first released about 18 months ago, and it now runs much smoother and has more features than the version that appeared on a QL-Today cover disk in 2004.
From comments during this session it is clear there are some QL-ers who are impressed by QDT, but who feel it is not for them. I suspect there is a fear that QDT could take over their machine in the sense that Windows takes over the PC. Paradoxically this is a compliment to the high technical and visual quality of program. Indeed some said the new File Manager has the potential to make QPAC2 obsolete.

No report of a Hove show is complete without a comment on the catering. This is something we are beginning to take for granted. All QL shows have built up a good reputation for the high standard and the cheap price of the catering. It is perhaps time to remember that catering is not just a matter of putting a kettle on for tea and coffee, but requires detailed logistical planning and some risks in estimating attendance. As usual the ladies of Hove did an excellent and much appreciated job.


[The QL Today team is wondering who Ken is planning to attack with his banana.]

At the moment no further UK shows are planned, but Ken Bain was present at Hove and we did a bit of arm twisting. The last two Byfleet shows have been disappointing with a low attendance of both traders and punters. In these circumstances an organiser of a show will naturally question if all the effort is worthwhile, but Ken promised to explore the possibility of a London or Byfleet show in the Autumn.

Manchester Postscript

My apologies to Tony Firshman for describing him as "singing innocently in his church choir" in my report of the Manchester show in the last issue of QL-Today.
Tony assures me he always sings innocently, but on that occasion it was in his chamber choir.

John Gilpin posted a reaction to the piece on the QL-users group:
"I can't understand your comment about Manchester always being a difficult location for a show. It has a brilliant transport network of motorways, long and short distance bus routes and a high speed tram service. It is the very hub of the rail network and the International Airport is well within striking distance of the chosen venue (Via its Railway Station where they serve early morning coffees and bacon butties to die for!! - to say nothing of the available refreshment houses in the locality especially those who look after their drinking glasses but who turn a blind eye to Grand Larceny of the table linen). I think that if our attendees did not have an overnight stay - considering that it was a TWO day event, - then the attendance would have been even worse.
All in all a very fair and thorough report of the event and, following our return from holiday, my renewal subscription is winging - or has winged - its way to QL Today headquarters so that I can be assured of access to the next thrilling instalment - The Hove Workshop."

Programming in Assembler – Part 15 - Dataspace problems

by Norman Dunbar

There has been much correspondance recently (today is the 3rd of April 2006) on the ql-users email list since Dilwyn posted a messages about the seeming "Catch-22" problems where someone downloads a QL application from the internet and has to extract the zip file on a Windows PC. The files thus extracted are then read into QPC (or similar) and subsequently, any executable files fail to work.

The problem is caused by the need for QL files to have a correct file header which has the file type byte set to be exectable and a valid value in the data space part of the header.

Getting hold of a QL specific version of Zip/Unzip is qute simple, but it arrives in a zip file so we have a recursive problem here. Dilwyn's advice is quite simple, load the program into memeory and SEXEC it with the correct data space. This is indeed a simple solution, but I wouldn't be writing this article if I didn't have an alternative would I?

Actually, there is a version of Zip for the QL which has been converted to run as an exe file that will extract itself, so this is the easiest solution overall. However, the utility I describe below can be used to make any file executable and to provide a data space value without having

QUANTA

Independent QL Users Group

World-wide Membership is by subscription only,
Offering the following benefits:
Bimonthly Newsletter – up to 40 pages
Massive Software Library – All Free!
Free Helpline and Workshops
Regional Sub-Groups. One near you?
Advice on Software and Hardware problems
Subscription just £14 for UK members
Overseas subscription £17

Barclaycard: Visa: Access: MasterCard: Accepted

Now in our Twenty Third Year

Further details from the Membership Secretary

John Gilpin, 181, Urmston Lane
Stretford, Manchester, M32 9EH (UK).
Tel. +44 (0) 161 865 2872
Or
Visit the Quanta Web Site
http://www.quanta.org.uk
E-mail: membership@quanta.org.uk

Next QUANTA Sponsored Events

North American QL Users Group
are holding
The North American QL Show
At
Super 8 Motel, Niagara Falls
(Canadian side)
Saturday September 30th. 2006
10.00 am to 4.00 pm
With a Group Dinner on the
previous evening at 6.00 pm.
Full details from bcable@coat.com

SURREY QUANTA SUB-GROUP AND LONDON QL AND QUANTA GROUP
PRESENT

BYFLEET '06 QUANTA WORKSHOP

ON SUNDAY 5 NOVEMBER 2006, FROM 10AM TO 4 PM
IN BYFLEET VILLAGE HALL, SURREY

(WWW.SADEYE.CO.UK FOR LATEST INFO)

to read the file off disc, delete it and then SEXEC a new copy back to the disc - what happens if you have a crash just after the delete?

The code below is based on a utility I wrote many years ago (1991 actually) that was supplied with WinBack when it was still a commercial program. I have only had to make a slight change to it for this version.

In the old days, the utility checked to make sure that the file was already executable and failed if not, this version doesn't fail, it simply changes the file to be executable.

When you EXEC the dataspace_bin file, you will be prompted for a filename. Type one in and if it is not an executable already, the program will advise you of this - then convert it to an EXEC file.

Next you will be shown the current data space and asked for a new value. The value you type should be numeric (or numeric with a 'k' at the end) and even. If not even it will be rounded up to the next even number.

If the value you type is less than the minimum the program allows (default is 1024 bytes), then your value will be rounded up to the minimum value. If you don't like this, then set minimum to zero in the source code and it will be ignored.

To finish the program, simply press ENTER when prompted for a filename.

We shall begin the typing with the usual set of equates and constants, type the following into a file - I suggest you name it dataspace_asm, but this is not mandatory.

```
*=====================================================================*
* DATASPACE version 1.10                                              *
*                                                                     *
* Copyright Norman Dunbar February 1991/2006                          *
*                                                                     *
* Changes a task file's dataspace.                                    *
*=====================================================================*
*                                                                     *
* AMENDMENTS                                                          *
*                                                                     *
* 03/04/06 - now makes files executable if not and doesn't complain.  *
*=====================================================================*

*---------------------------------------------------------------------*
* EQUATES  General                                                    *
*---------------------------------------------------------------------*
ftyp       equ     $05                 Offset to file type
fdat       equ     $06                 Offset to file dataspace
exec_file  equ     $01                 Indicator that file can be EXEC'd
minimum    equ     1024                Minimum dataspace allowed

me         equ    -$01                 This job
linefeed   equ     $0A                 Ascii line feed
timeout    equ    -$01                 Infinite timeout
black      equ     $00                 Black ink/paper code
red        equ     $02                 Red ditto
green      equ     $04                 Green ditto
white      equ     $07                 White ditto
```

Following on from the above, we have the standard QDOSMSQ job header. From the code that follows below, you can see the header with the job name and version in the usual format for a QDOSMSQ job.

```
*=====================================================================*
* The code starts here with a standard QDOS job header.               *
*=====================================================================*
start      bra.s   dataspace           Jump over header
           dc.l    0                   Make sure $4AFB is at offset 6
           dc.w    $4AFB               ID word
```

```
name       dc.w    22                  Length of name
           dc.b    'Dataspace Version 1.10'

*-------------------------------------------------------------------------*
* Open a console window                                                   *
*-------------------------------------------------------------------------*
dataspace  move.w  ut_con,a2
           lea     con_def,a1          Console definition block
           jsr     (a2)                Open a CON_ channel
           tst.l   d0                  Check for errors
           bne     job_end             And bale out if found
           move.l  a0,d7               Store console id

*-------------------------------------------------------------------------*
* Console is open, sign on                                                *
*-------------------------------------------------------------------------*
sign_on    lea     name,a1             Job name
           bsr     write_text          Print job name
           lea     copyright,a1        Copyright message
           bsr     write_text          And copyright message
```

After the job header, the first part of the program performs all the initialisation that is required. The job opens a new console channel and saves the channel number in D7.L. D7.L is not corrupted by any of the code that follows, so it is a good place to save the channel number. It will be used each time we pass through the main loop.
It is slightly more efficient to move data between two registers than it is to fetch from a memory location. In this utility, that's hardly going to be needed, but we might as well use all the registers before we have to start saving data in memory.
Once the main console channel is open, we print a small sign on message showing the job name (extracted from the standard job header area) and a copyright message and then drop into the main processing loop.
Please note that although this code is copyright, you have my express permission to use and abuse it as you see fit. If the code can be useful in programs that you write in future, please feel free to copy it directly with my blessings!
You will note that much of this program is written as simple subroutine calls.
Once again, reusing existing and working code is always a good idea in my book.

```
*-------------------------------------------------------------------------*
* Main loop, keep asking for a filename until ENTER only pressed          *
*                                                                         *
* First prompt for filename                                               *
*-------------------------------------------------------------------------*
main_loop  move.l  d7,a0               Console id
           lea     mess_1,a1
           bsr     write_text          Enter filename message
           bsr     get_text            Get filename
```

The main loop starts off by getting the console channel number into A1 which is where most (if not all) the channel handling routines expect it to be. D7 is never corrupted by the code below so makes a good place to save it.
The user is prompted to 'enter a filename or press ENTER only to quit', the program waits for a response from the user and if the user simply pressed ENTER, the program will skip off to the code at label 'any_key' to terminate the program.

```
*-------------------------------------------------------------------------*
* Then check if it was only ENTER and if so exit the job                  *
*-------------------------------------------------------------------------*
check_end  tst.w   d1                  Finished ?
           beq     any_key             Yes
```

```
*------------------------------------------------------------------------*
* Otherwise attempt to open the file                                     *
*------------------------------------------------------------------------*
open_file  moveq   #io_open,d0        Open file
           moveq   #me,d1             For this job
           moveq   #0,d3              For input
           move.l  a1,a0              File name is in buffer
           trap    #2
           tst.l   d0                 Check errors
           beq.s   read_head          Open was ok

*------------------------------------------------------------------------*
* Cannot open the file, print its name and the error message             *
*------------------------------------------------------------------------*
cant_open  move.l  d0,-(a7)           Store error code
           lea     mess_2,a1          Cannot open message
           bsr     write_text         Print it
           lea     input,a1           File name
           bsr     write_text         Print filename
           lea     mess_6,a1          A colon
           bsr     write_text         Print it
           move.l  (a7)+,d0           Get error code
           move.w  ut_err,a2
           jsr     (a2)               Print error message

*------------------------------------------------------------------------*
* Note, D0 is preserved by UT_ERR, so cannot check for errors            *
*------------------------------------------------------------------------*
           bra.s   main_loop
```

Assuming that we got a response back from the user, we assume that it is a filename which the user wishes to either make executable, or adjust the data space. The code at 'open_file' above attempts to open the filename supplied by the user for input. This means that it must already exist on a device somewhere.
If the file opened ok, the program skips to the code below which attempts to read the file header, otherwise the user is presented with a message detailing why the file couldn't be opened, and we skip back to the start of the main loop where we prompt for another filename.
You will note my comment that UT_ERR preserves the error code passed to it in D0, so we cannot make any checks to determine whether the error code in D0 is the one we passed in, or one generated by UT_ERR. Don't worry about it :o)
A quick tangent is requred now, before we proceed. On a directory device such as 'flp1_', 'mdv1_' and so on, all files have two parts to them. First of all there is the file header - a 64 byte section of data which is stored in the directory (and allegedly at the start of each file too - you just can't get at it!) - and the contents of the file proper.
This header holds details of the file size, it's type, data space, various dates giving details of when the file was last changed, backed up etc. We are interested in only two parts of the header in this utility - the file type (byte) and the file's data space (long).
We defined a couple of equates way back at the start of the code - the 'ftyp' equate points at byte 5 of the 64 byte buffer and the 'fdat' points at the long word starting at byte 6 of the header. These are the two places we need to get data from and write data to.

On with the real code again. The following reads a file header and processes errors as required.

```
*------------------------------------------------------------------------*
* File has been opened ok, read the file header                          *
*------------------------------------------------------------------------*
read_head  move.l  a0,d6              Store file id
           moveq   #fs_headr,d0
           moveq   #64,d2             Size of buffer
```

```
          moveq   #timeout,d3         Timeout
          lea     buffer,a1           Put header here
          move.l  a1,a5               Store buffer
          trap    #3                  Go get the file header
          tst.l   d0                  Check for errors
          beq.s   check_exec          none

*---------------------------------------------------------------------*
* Cannot read file header, say so and print the error message         *
*---------------------------------------------------------------------*
cant_read lea     mess_3,a1           Cannot read header message
          move.l  d0,-(a7)            Store error code
          bsr     write_text          Print message
          move.l  (a7)+,d0            Get error code
          move.w  ut_err,a2
          jsr     (a2)                Print error message
          bra     main_end            Skip the rest of the loop
```

The code above tries to read the 64 byte file header into a buffer - which must be big enough to hold all 64 bytes - and if an error is detected in the attempt, the user is told about the problem and the file is closed prior to the main loop starting all over again.
The buffer start address is saved in register A5.L for later use prior to the attempt to read the file header - this is because the real buffer address register, A1.L will be corrupted by the trap call.

```
*---------------------------------------------------------------------*
* Header has been read ok, check if the file is EXECable              *
*---------------------------------------------------------------------*
check_exec cmpi.b #exec_file,ftyp(a5) Check file is EXECable
          beq.s   current             It is

*---------------------------------------------------------------------*
* File is not EXECable, print a warning message and convert it        *
*---------------------------------------------------------------------*
not_exec  lea     input,a1            Filename
          bsr     write_text          Print filename
          lea     mess_4,a1           Not an EXECable file message
          bsr     write_text          Print the message
          move.b  #exec_file,ftyp(a5) Make file EXECable

*---------------------------------------------------------------------*
* File is EXECable, print its current dataspace                       *
*---------------------------------------------------------------------*
current   lea     mess_5,a1           Current dataspace is message
          bsr     write_text          Print it
```

The file is now open, its header has been read into our buffer. The code now checks to see if this file is executable already. If it is, nothing is said or done, howver, if the file is not executable - and this will be the case for a file extracted by a Windows version of Zip - we display a warning message to the user and convert the file to be EXECable.
In either case, we print a message which will eventually inform the user how big the file's dataspace is at the moment. The first part of the message is easy - it is simple text but we also need to print out the current value. This is done by the code below.

```
*---------------------------------------------------------------------*
* Now get the current dataspace & convert it to ASCII                 *
*---------------------------------------------------------------------*
          move.l  fdat(a5),d3         D3.L is dataspace
          lea     input+2,a1          A1.L is output buffer
          lea     tens_table,a2       Powers of 10
          moveq   #0,d1               D1.W is digit counter (all digits)
```

```
next_digit move.l  (a2)+,d2             D2.L is current power of 10
           beq.s   all_done             But zero is end of table
           clr.b   d0                   D0.B is current digit

digit_loop sub.l   d2,d3                Subtract the current power of 10
           blt.s   buff_digit           Too far
           addq.b  #1,d0                Increase current digit
           bra.s   digit_loop           And try again

buff_digit add.l   d2,d3                Correct for the overflow
           tst.b   d0                   Is this a zero ?
           bne.s   not_a_zero           No
           tst.w   d1                   Yes, is it a leading zero ?
           beq.s   next_digit           Yes, ignore it

not_a_zero addi.b  #'0',d0              Convert to ASCII
           move.b  d0,(a1)+             Store in buffer
           addq.w  #1,d1                Increment total digits
           bra.s   next_digit           And do the rest

*-----------------------------------------------------------------------*
* Check for a result of zero. In this case force a '0' to be printed    *
*-----------------------------------------------------------------------*
all_done   tst.w   d1                   Any digits found ?
           bne.s   not_zero             yes
           move.b  #'0',(a1)            Store a zero
           moveq   #1,d1                And set the count

not_zero   lea     input,a1             The buffer
           move.w  d1,(a1)              Store character count

*-----------------------------------------------------------------------*
* Dataspace is converted, print it out                                  *
*-----------------------------------------------------------------------*
           bsr     write_text           Print old dataspace
```

The code above begins by reading the long word representing the file data space requirements from the file header into register D3.L.
D3.L is then converted to text read for printing by the fairly simple method of repeatedly subtracting assorted powers of 10 from the value until we get an overflow (underflow?) and saving the number of times we managed to sucessfully subtract the current power of 10. This count is our current digit and is held in D0.B as it cannot be greater than 9.
If the counter is non-zero, we convert it to ASCII by adding the ASCII code for '0' (zero) to the count and save it in the buffer located at A1. We only print out the full number when we have decoded it - we don't print each digit individually as we go along.
If the dataspace is still zero, even after processing all the digits, we simply print a zero.

```
*-----------------------------------------------------------------------*
* Now prompt for, and read in the required new dataspace                *
*-----------------------------------------------------------------------*
new        lea     mess_8,a1            New dataspace message
           bsr     write_text           Print it
           bsr     get_text             Get new dataspace
           tst.w   d1                   No text ?
           beq.s   new                  Try again
           move.w  d1,d0                Get text length
           subq.w  #1,d0                Adjust for dbra
           addq.l  #2,a1                Adjust A1 past the word counter
```

Having printed out the current data space for the file in question, we next prompt the user to enter a new value. If the user simply presses ENTER without entering a value, the program detects this, and simply loops around to ask for the new data space value again. To get out of this loop a valid numeric value must be entered.
The utility accepts pure digits or a number of digits suffixed by a 'K' (in upper or lower case) and this is used to specify a data space in Kilobytes rather than bytes. If there are spaces in the user input, they will simply be skipped.

```
*---------------------------------------------------------------------*
* Convert from ASCII into binary, ignore leading (any) spaces and stop *
* if a 'K' or 'k' is detected. Reject all other non-digit characters  *
*---------------------------------------------------------------------*
convert   moveq   #0,d4           Needs to be a long word
          move.l  d4,d5           D5 is total so far

conv_next move.b  (a1)+,d4        Get a byte
          cmpi.b  #' ',d4         Is it a space ?
          beq.s   try_next        Yes, ignore it
          cmpi.b  #'k',d4         Is it 'k'
          bne.s   try_K           no

mul_1024  asl.l   #2,d5           Yes, multiply by 1024
          asl.l   #8,d5           Can't do it in one go
          bra.s   make_even       And exit

try_K     cmpi.b  #'K',d4         Try uppercase
          beq.s   mul_1024        Yes

          cmpi.b  #'0',d4         Is it a digit ?
          bcs.s   invalid         No
          cmpi.b  #'9',d4         But it might be
          bls.s   mul_10          Yes it is

*---------------------------------------------------------------------*
* An invalid digit has been detected, print error message & try again *
*---------------------------------------------------------------------*
invalid   lea     mess_10,a1      Invalid digit message
          bsr.s   write_text      Print it
          bra.s   new             try again
```

As we scan through the input supplied by the user we ignore any spaces. I could have written the code to detect when the first digit had been detected and processed spaces after that as errors, but I was obviously too lazy to do so back in 1991. (Not much has changed in 2006 then!)
Each character is checked and if it is a 'K' (in any letter case) it indicates the end of the input and the current value is multiplied by 1024 to get the correct number of bytes.
If an invalid character is detected, an error message is printed and the user restarts the inner loop of the program where s/he is prompted to type in a new data space value.

```
*---------------------------------------------------------------------*
* Multiply D5.L by 10 and add in the digit just read                  *
*---------------------------------------------------------------------*
mul_10    asl.l   #1,d5           D5 = D5 * 2
          move.l  d5,d3           Store for now
          asl.l   #2,d5           Now D5 = D5 * 8
          add.l   d3,d5           And finally D5 = D5 * 10
          subi.b  #'0',d4         Convert byte to (long) binary
          add.l   d4,d5           Total = (total * 10) + digit

try_next  dbra    d0,conv_next    Do rest of digits
```

If we get to the code above, then the current character in the user input must be a digit. We multiply the running total so far by 10, convert the latest digit from an ASCII character down to a numeric value and add it to the running total in D5.L before skipping back to continue scanning the input area for another digit.

```
*---------------------------------------------------------------------*
* When finished, the value must be even                               *
*---------------------------------------------------------------------*
make_even  addq.l  #1,d5              Prepare to make even
           bclr    #0,d5              Make dataspace even

*---------------------------------------------------------------------*
* And, not less than minimum allowed                                  *
*---------------------------------------------------------------------*
           cmpi.l  #minimum,d5        Check new size
           bcc.s   set_head           It's ok.
           move.l  #minimum,d5        Make sure it is = minimum size
```

Eventually, we exit from the loop scanning the user's input and arrive at the code above. This is a short but very important piece of code. The data space for a file must be even. If it is odd, then any attempt to EXEX (EX etc) the code will result in an address exception with the usual resulting lock up. Just say no to odd addresses!
The running total in D5.L is rounded up to the next even number, or left alone if it is already even.
If the running total is less than our minimum allowed value, it is set to that value.

```
*---------------------------------------------------------------------*
* Now load the header with the new dataspace and set it               *
*---------------------------------------------------------------------*
set_head   move.l  d5,fdat(a5)        Store in the header
           moveq   #fs_heads,d0       Set the file header
           moveq   #timeout,d3        Timeout
           move.l  d6,a0              File id
           move.l  a5,a1              File header
           trap    #3                 Go set it
           tst.l   d0                 Any errors ?
           bne.s   not_set            Yes
```

As we now have a new data space value in D5.L, we save it in the file header buffer in the correct location and call the QDOSMSQ trap call to write the file header back to the device. If this works ok, we drop into the following code to flush the changes to the device.
QDOSMSQ is happy to save changes in the file slave area until it has a moment to write them out. This is ok in most cases, but if a user wishes to remove a floppy disc, for example, that we must make sure that all changes are written down to the disc. The following code does that task, closes the file and starts the main loop all over again.

```
*---------------------------------------------------------------------*
* Now flush out the file buffers, so that if I change discs I have    *
* written the header to the current disc. Cannot detect the QDOS error *
* READ/WRITE failed (try removing the disc and it won't fail or       *
* produce an error code). It might print a message if it can find an  *
* open command channel which is not in use. I got it when testing via *
* a monitor but not while running on its own.                         *
*---------------------------------------------------------------------*
flush      moveq   #fs_flush,d0       Prepare to flush the buffer
           moveq   #timeout,d3        This could take all day
           trap    #3                 But do it anyway
           tst.l   d0                 And check for errors
           beq.s   main_end           None, do the next file
```

RWAP Services NEWS!

We have moved back to Walsall for the time being - see our current address at the bottom of the page.

Better news is that we now have an extended range of second hand items for the Sinclair QL and ZX Spectrum, and have one remaining Epson 850 Inkjet printer available. Z88 stocks are however now getting quite low so if there is anything you need, please browse our website:

http://www.rwapsoftware.co.uk

We are also looking to produce some new hard disk interfaces for the ZX Spectrum and have a few little projects on the drawing board.

Our websites:
http://www.rwapservices.co.uk (General site)
http://www.rwapsoftware.co.uk (Sinclair computer second hand and new items)
http://www.rwapadventures.com (Adventure Programs)
http://www.internetbusinessangels.com (Guidance on setting up online businesses).

New Products!

QWORD 1.0 NOW WITH DIGITAL SOUND ON QPC2!

The wait is now over! Q-Word version 1 is finally available!

Platforms:
QPC/QXL, Q40/Q60, Aurora (with SGC)

Prices:

All versions without P-Word	£20.00
All versions with P-Word	£30.00

Notes:
Q-Word DOES NOT require SMSQ/E with GD2 support -OR- SMSQ/E at all on the Aurora or Qx0 machines. It works on the highest colour depth everywhere regardless of Operating System.
The Aurora version is available on either HD or ED disk. For the latter add £1.00 to the price. ED version is uncompressed and can be run directly from the floppy. All other Floppy versions are compressed. QPC/QXL version comes on CD. Non CD versions DO NOW support digital sound on QPC2

Quantum Leap ED **Drives**

The bad news is that our stock of ED Disk Drives has now been depleted and there is no sign of any more being available in the short term.

We do however have a range of brand new DD and ED 3.5" Diskettes for sale at low prices:

10 x 3.5" DSDD Disks £7.50
10 x 3.5" ED Diskettes £10
10 x 3" Disks £9

Prices do not include post and packing - please ask us for details.

We do have a range of second hand disk interfaces and drives (one or two ED Drives) for use with the Sinclair QL, so if you need anything, please let ask.

TALENT COMPUTER SYSTEMS for **Windows**

For QLers that run Windows or with incompatible hardware for Talent Games, we now have re-released these adventures so that they can run on your Windows-equipped PC. No Emulator, floppies, microdrive backups etc. required, just a one-click install! Of course the full QL line is still available! (See side column)

Talent Games for Windows ea. £ 10.00
(Each Game includes a runtime installation of QLAY-2 by Jimmy Montesinos)

Old Favourites!

Utilities

SBASIC / SuperBASIC Reference Manual on CD	**£ 20.00**
Sidewriter v1.08 *Landscape Printing (EPSON printers)*	£ 10.00
ImageD v1.03 *3D object generator*	£ 10.00
Q-Help v1.06 *Superbasic On-Screen help system*	£ 10.00
Q-Index v1.05 *Keyword-to-topic finder*	£ 5.00
ProForma ESC/P2 Drivers v1.04 for ProWeSs *Printer Driver*	£ 8.00

Applications

Flashback SE v2.03 (upgrade only) *Database*	£ 5.00
QL Cash Trader v3.7 *Accounting/Finance*	£ 5.00
QL Payroll v3.5 *Accounting/Finance*	£ 5.00
QL Genealogist v3.26 *Genealogy*	£ 20.00
Genealogy for Windows	£ 50.00
QL Genealogist to Windows version upgrade	£ 25.00
QL Cosmos v2.04 *Planetarium*	£ 5.00
Q-Route v2.00 *Route Finding*	£ 25.00
Upgrade from v1.xx	£ 5.00
Britain map v1.11	£ 2.00
BIG Britain map (needs 2Mb) v2.03	£ 5.00
Various Britain Area maps (ask for details)	ea. £ 2.00
Ireland map v1.00	£ 5.00
Belgium map v1.01	£ 2.00
Catalonia map v1.03	£ 2.00
P-Word UK English Dictionary (500.000 words!) *Dictionary*	£ 15.00

Leisure

Return to Eden v3.08 *Adventure*	£ 10.00
Nemesis MkII v2.03 *Adventure*	£ 8.00
The Prawn v2.01 *Adventure*	£ 8.00
Horrorday v3.1 *Adventure*	£ 8.00
West v2.00 *Adventure*	£ 5.00
The Lost Kingdom of Zkul v2.01 *Adventure*	£ 5.00
All 6 games above	£ 25.00
D-Day MkII v3.04 *Strategy/War Simulation*	£ 10.00
Grey Wolf v1.08 *Graphical Submarine Simulation*	£ 8.00
War in the East MkII v1.24 (upgrade only) *Strategy/War Simulation*	£ 5.00
Open Golf v5.20 *Sports Simulation*	£ 8.00
QuizMaster II v2.07 *Quiz*	£ 5.00
Stone Raider II v2.00 *Arcade Game*	£ 5.00
Hoverzone v1.2 *Arcade Game*	£ 5.00
Deathstrike v1.5 *Arcade Game*	£ 5.00
Flightdeck v1.0 *Flight Simulation*	£ 10.00
All 6 games above (Open Golf, QuizMaster II, Stone Raider II, Hoverzone, Deathstrike and Flightdeck)	£ 28.00

Notes on Software requirements
The following programs have a minimum SGC card requirement: P-Word, Qword, Big Britain MAP for Q-Route

RWAP Services
26 Oak Road, Shelfield, Walsall, West Midlands WS4 1RQ TEL: +44 1922 691607 From the UK Dial: 01922 691607
Website: http://www.rwapsoftware.co.uk
Email: sales@rwapsoftware.co.uk

We Accept Payment using:
PayPal VERIFIED NOCHEX EMAIL MONEY PREFERRED PAYMENT (Cheques in £ sterling made payable to R. Mellor)

If the file header failed to be written, the following code will inform the user that there is a problem and display the QDOSMSQ error message.

```
*-------------------------------------------------------------------*
* Header not set or flush failed, print error message               *
*-------------------------------------------------------------------*

not_set     move.l  d0,-(a7)            Store error code
            lea     mess_7,a1           Cannot set header message
            bsr.s   write_text          Print it
            move.l  (a7)+,d0            Get error code
            move.w  ut_err,a2
            jsr     (a2)                Print error message

*-------------------------------------------------------------------*
* Can't trap errors in UT_ERR as D0 is preserved.                   *
* Close the file & loop to the start                                *
*-------------------------------------------------------------------*
main_end    move.l  d6,a0               File id for close
            moveq   #io_close,d0
            trap    #2                  Close the file
            bra     main_loop           And see if more to be done
```

We have now reached the end of the main loop. The code above retrieves the file's channel number from register D6.L, closes the file and skips back to the start of the main loop ready for the next file to be processed.
The code below is a collection of simple subroutines, some you will have seen before, which carry out various useful parts of the program. The comments above each should be sufficient to explain what is going on.
Also included below are some data input and header buffer areas.

```
*-------------------------------------------------------------------*
* Subroutine to print text to screen                                *
*                                                                   *
* ENTRY                                                             *
*                                                                   *
* D7.L = Channel id                                                 *
* A1.L = Pointer to text to print (Word then bytes)                 *
*                                                                   *
* EXIT                                                              *
*                                                                   *
* A0.L = channel id                                                 *
*-------------------------------------------------------------------*
write_text move.w  ut_mtext,a2         Print text vector
           move.l  d7,a0               Channel id
           jsr     (a2)                Print it
           tst.l   d0                  Check errors
           bne.s   job_error           Oops kill job
           rts                         Otherwise exit

*-------------------------------------------------------------------*
* A fatal error has occurred, print it, wait for any key and kill job *
* wait for a key allows WMAN & PTR_GEN users to see the message before *
* WMAN restores the screen.                                         *
*-------------------------------------------------------------------*
job_error  move.l  d7,a0               Get console id
           move.w  ut_err0,a2          Print error text vector
           jsr     (a2)                Print to #0

any_key    move.l  d7,a0               In case entry is here
           lea     mess_12,a1          Press any key message
           move.w  ut_mtext,a2         Don't use WRITE_TEXT in case of errors
```

```
         jsr     (a2)                Print it
         moveq   #io_fbyte,d0        Fetch one byte
         moveq   #timeout,d3         Take all day if you like
         trap    #3                  Go get it
         moveq   #io_close,d0
         trap    #2                  Close console channel

*------------------------------------------------------------*
* This job will self destruct in no time at all              *
*------------------------------------------------------------*
job_end  moveq   #mt_frjob,d0        Job is about to die
         moveq   #me,d1              And it is this job
         trap    #1                  RIP (there is not return)

*------------------------------------------------------------*
* Subroutine to get some text from the user                  *
*                                                            *
* ENTRY                                                      *
*                                                            *
* D7.L = channel id                                          *
*                                                            *
* EXIT                                                       *
*                                                            *
* D1.W = number of bytes read                                *
* A0.L = channel id                                          *
* A1.L = start of buffer (word then bytes)                   *
*------------------------------------------------------------*
get_text lea     input,a1            Buffer for the text
         move.l  a1,-(a7)            Store it
         addq.l  #2,a1               Leave room for the length word
         moveq   #io_fline,d0
         moveq   #42,d2              Maximum buffer size
         moveq   #timeout,d3         Take as long as you like
         trap    #3                  Get some text
         tst.l   d0                  Check for errors
         bne.s   job_error           Bale out (stack will be ok)
         move.l  (a7)+,a1            Get buffer start
         subq.w  #1,d1               Remove the line feed
         move.w  d1,(a1)             Store text length
         rts                         Exit

*------------------------------------------------------------*
* Definition block for my console channel                    *
*------------------------------------------------------------*
con_def  dc.b    black               Border colour
         dc.b    $01                 Border width
         dc.b    white               Paper & strip colour
         dc.b    black               Ink colour
         dc.w    $01C0               Width = 448
         dc.w    $0064               Height = 100
         dc.w    $0020               X position = 32
         dc.w    $0010               Y position = 16

*------------------------------------------------------------*
* Copyright message, so the world knows my name              *
*------------------------------------------------------------*
copyright dc.w   copy_end-copyright-2
         dc.b    linefeed
         dc.b    'Copyright Norman Dunbar, Jan 1991/April 2006.'
         dc.b    linefeed
copy_end equ     *
```

```
*-------------------------------------------------------------------------*
* Various prompts & error messages                                        *
*-------------------------------------------------------------------------*
mess_1     dc.w    end_1-mess_1-2
           dc.b    linefeed
           dc.b    'Enter filename'
           dc.b    linefeed
           dc.b    'or ENTER only to finish : '
end_1      equ     *

mess_2     dc.w    end_2-mess_2-2
           dc.b    'Cannot open '
end_2      equ     *

mess_3     dc.w    end_3-mess_3-2
           dc.b    'Cannot read file header : '
end_3      equ     *

mess_4     dc.w    end_4-mess_4-2
           dc.b    ' is being converted to an EXECable file'
           dc.b    linefeed
end_4      equ     *

mess_5     dc.w    end_5-mess_5-2
           dc.b    'Current dataspace is : '
end_5      equ     *

mess_6     dc.w    end_6-mess_6-2
           dc.b    ' : '
end_6      equ     *

mess_7     dc.w    end_7-mess_7-2
           dc.b    'Cannot set file header : '
end_7      equ     *

mess_8     dc.w    end_8-mess_8-2
           dc.b    linefeed
           dc.b    'Enter new dataspace in bytes, or'
           dc.b    linefeed
           dc.b    'end with "K" for kilobytes : '
end_8      equ     *

mess_9     dc.w    end_9-mess_9-2
           dc.b    ' bytes'
           dc.b    linefeed
end_9      equ     *

mess_10    dc.w    end_10-mess_10-2
           dc.b    'Invalid digit found in input'
           dc.b    linefeed
end_10     equ     *

mess_11    dc.w    end_11-mess_11-2
           dc.b    'Dataspace set.'
           dc.b    linefeed
end_11     equ     *

mess_12    dc.w    end_12-mess_12-2
           dc.b    linefeed
           dc.b    'Goodbye, press any key to kill job.....'
end_12     equ     *
```

```
*-----------------------------------------------------------------------*
* Two buffer areas, one for the file header & one for user input        *
* note how sneaky I have been, by using DS.W I have forced them both    *
* to be word aligned. If I had used DC.B they might not have been, and *
* I would be bound to get an address exception sometime. (it happened) *
*-----------------------------------------------------------------------*

buffer     ds.w    32                  Buffer is 64 bytes maximum
input      ds.w    22                  Size = 41 + ENTER + word count

*-----------------------------------------------------------------------*
* A table of all powers of ten, from 9 to 0. This corresponds to the    *
* values used when converting an UNSIGNED long word to ASCII.           *
* 2^31 = 2,147,483,648                                                  *
* 10^9 = 1,000,000,000 so is a big enough 'highest' power to use        *
*-----------------------------------------------------------------------*
tens_table dc.l    1000000000          10 ^ 9
           dc.l    100000000           10 ^ 8
           dc.l    10000000            10 ^ 7
           dc.l    1000000             10 ^ 6
           dc.l    100000              10 ^ 5
           dc.l    10000               10 ^ 4
           dc.l    1000                10 ^ 3
           dc.l    100                 10 ^ 2
           dc.l    10                  10 ^ 1
           dc.l    1                   10 ^ 0
           dc.l    0                   Table end marker
```

Wildcards

by Dilwyn Jones

I've raided my library of routines for this little program. This routine allows me to use the common '?' and '*' wildcard characters in string searches. It lets me do 'vague' searches of text strings or filenames.

? - this matches with any single character
* - this matches any length of text

The term 'wildcard' probably comes from card games where the Joker card can be used as any card the holder chooses.
Two of the most obvious applications are in word games where you can search for words where you only know part of the spelling, and filename pattern matching.

1. Word searches. Suppose you are doing a crossword, and know only 3 of the 6 letters in a word. You have a word list to hand (people like Geoff Wicks have ported word lists to the QL). You know that the first letter is C, the third letter is M and the sixth letter is N. So, if your program could search for C?M??N it would probably find the wordCOMMON in the list.

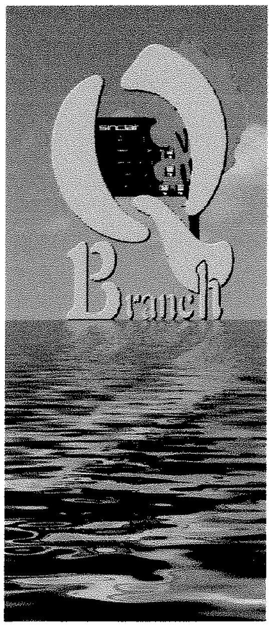
2. Filename patterns. Now that we have large hard disks and a huge number of programs all with different file types, it would be useful to be able to list only files ending with _DOC, for example. OK, you can use a DIR FLP1_ _DOC command in Superbasic, but this program will help you do something like DIR FLP1_*_DOC

It's written as a function which returns 1 if a specified search string is found within a given second string:

```
PRINT WILDCARD(wcard$,find$)
```

returns 1 if wcard$ is contained within find$, or 0 if not. wcard$ can contain one or more '?' or '*' wildcard characters, although it won't work if these are next to each other (pretty pointless if they were really).

Examples:
```
PRINT WILDCARD('C*N','COMMON')
```
returns 1 - this search tests for a word starting with C and ending with N. Any number of characters may be in between.



Feeling out on a Limb?
Reach out for QBranch
Suppliers of Quality QDOS/SMSQ products
Hardware and Software

20 Locks Hill
Portslade
Sussex
BN41 2LB
UK
Tel: +44(0)1273 486030
Fax: +44(0)1273 430501
Mobile: +44(0)7836 745501

email : sales@qbranch.demon.co.uk
www.qbranch.demon.co.uk

We can accept payment by Visa, Mastercard, Maestro and Switch. We also accept sterling cheques drawn on a UK bank. or payment direct to our bank:
Natwest
Sort code 60-17-01
Acc. 84534826
BIC : NWBKGB 2L
IBAN :
GB34 NWBK 6017 0184 5348 26

VISA MasterCard Maestro

QDT

The Ultimate QL Desktop
v1.02
£41.50 + p&p
£2.00 UK / 2.50 EU / £ 3.00 RoW
Needs SMSQ/E & extended colours to run

EasyPTR v4

Pts 1&2
Major Rewrite!
£42.00
Produce High Colour pointer-driven menus from SuperBASIC
Also on CD Rom

Text 87 £ 79.00

High Colour Patch (all systems) £9.00
ESC/P2 drivers £ 26.00 / Fountext £39.00
Typset94 / 2488 drivers £29.00 ea.

SMSQ/E and QPC 2

SMSQ/E for Gold Card / Atari / QXL / Q40
Aurora High Colour Driver included with Gold Card Version
£ 27.00

QPC 2 v 3.32
Now only £ 42.00

Upgrade Prices:
from v 3.xx Free
from v 2.xx £13.90
from v 1.xx £34.00

Delivered on CD ROM with Manual as Adobe Acrobat File

QPCPrint

Print to any Windows Installed Printer from Your QL Programs !
£27.00

Programs upgraded to high colour version

QMake / WinEd / QPAC 1 / QPAC 2 / QSUP / SuQcess / Disk Mate 5
Send old master disk and £1.60 per disk to cover costs

Free !

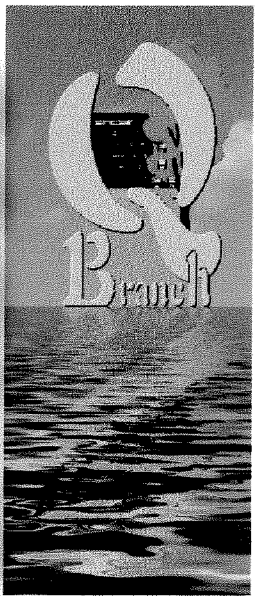
Paid Upgrades	
QD 98 to QD 2003	£10.50
QSpread 2001 to QSpread 2003	£10.50

Return Master Disk with *order*

We are having a bit of an pre-Autumn clearout here. the following items are offered at sale prices:
GAMES : QShang/Firebirds/Pipes/Mine Field Arcanoid (4/8 Colour only) £5.00 each + Postage
Black Knight Chess program / Super Games Pack £15.00 each

SUPER BARGAIN

MinisQL, Complete. Super Gold Card, Aurora, Full SuperHermes, 120Gb HDD, Qubide, 8mb RomDisq, SMSQ/E Only £250.00 Ready to go!



Programs

Utilities

Fifi2	File Finder	£21.00
QSup	Utilies	£30.00
QSpread	Spreadsheet	£51.00
Cueshell 2	File Manager	£15.00
QPAC 2	File Manager & Utilities Package	£42.00
QPAC 1	Calendar, Clock, Calculator, Sysmon	£22.00
QLoad/Qref	Fast load for QDOS	£15.00
QTYP2	Spell Checker	£31.00
QLQ	Printer Utility	£30.00
SuQcess	Database	£28.00
Q-Route	Route Finder	£25.00
Knight Safe3	Backup Program	£35.00
Fractal Collection	Fractals	£35.00
QCount	Accounting	£25.00

Programming

QD 2003	Text Editor & More	£ 49.00
QBASIC	QLiberater to QD Link	£ 15.00
QLiberator	Basic Compiler	£ 50.00
QD + QBASIC		£ 63.00
QD + QBASIC + QLiberator		£104.00
QPTR	Pointer Toolkit	£ 32.00
MasterSpy	Fast Text Editor	£ 30.00
QMake	Assembler Tools	£ 18.00
QMon/Jmon	Monitor - Upgrade only	£ 22.00
BASIC Linker	Basic Library Linker	£ 22.00
Disa 3	Dissassembler	£ 34.00
QMenu	Menu Extensions & tutorial	£ 16.00
Easyptr v4	Toolkits & Programming Extns	£ 41.50
Easyptr v4	Part 3 C extensions	£ 14.00

HARDWARE

We have a rotating stock of both new and second user hardware. It is best to call or email us for details of what is available.

Recycled Items
(when available)

Super Gold Card	£110.00
Gold Card	£ 45.00
Aurora	£ 65.00
QXL	£ 35.00
superHermes	£ 65.00
DiRen Keyboard Interface	£ 15.00
Qplane	£ 5.00

New Items

Aurora	£70.00
Aurora w/8301 & Minerva	£80.00
8mb Rom Disq	£98.00
4mb Rom Disq	£65.00
2mb RomDisq	£39.00
mPlane	£34.00
MCplate	£ 6.50
Varoius braQuets	£ 8.00
K'board Membrane	£17.50
Gold /Super Gold Card Batteries	£10,00

We also have a collection of standard QLs, QL Power supplies and some QL books.

Cables for the Aurora, Qubide and Super Gold Card ROMs and other QL accessories are also available from us.

Call for details

```
PRINT WILDCARD('*D','loaded')
```
returns 1 – searches for a word ending with D and any number of characters before the D.
```
PRINT WILDCARD('*N*','WILD')
```
returns 0 – there is no N in ZERO.
```
PRINT WILDCARD('?N*',z$)
```
returns 1 if the string in z$ contains N as the second letter. The ? character always stands for one character (which could be anything). * stands for any number fo characters, so what this command looks for is a word which contains N as its second letter, and any number of characters after the N. So
```
WILDCARD('?N*','another')
```
returns 1, but
```
WILDCARD('?N*','An')
```
returns 0 because the * implies there should be something else after the letter N no matter what it is.
```
PRINT WILDCARD('FLP1_*_DOC','FLP1_README_DOC')
```
returns 1.
```
PRINT WILDCARD('FLP1_*_ABA','FLP1_README_DOC')
```
returns 0.

Figure 1 - The Wildcard function listing

```
20000 DEFine FuNction Wildcard (wc$,search$)
20010   LOCal wc_ptr,sch_ptr
20020   IF wc$ = '' OR search$ = '' THEN RETurn 0
20030   sch_ptr = 1 : REMark running pointer along search$
20040   FOR wc_ptr = 1 TO LEN(wc$)
20050     IF sch_ptr > LEN(search$) THEN RETurn 0
20060     REMark handle the '?' wildcard character
20070     IF wc$(wc_ptr) = '?' THEN sch_ptr = sch_ptr + 1
20080     REMark handle the '*' wildcard character
20090     IF wc$(wc_ptr) = '*' THEN
20100       IF wc_ptr = LEN(wc$) THEN RETurn 1
20110       sch_ptr = sch_ptr + 1
20120       IF sch_ptr > LEN(search$) THEN RETurn 0
20130       REMark search for character following wildcard
20140       REPeat character_search
20150         IF wc$(wc_ptr+1) == search$(sch_ptr) THEN EXIT character_search
20160         sch_ptr = sch_ptr + 1
20170         IF sch_ptr > LEN(search$) THEN RETurn 0
20180       END REPeat character_search
20190     END IF
20200     REMark character other than wildcard
20210     IF wc$(wc_ptr) <> '?' AND wc$(wc_ptr) <> '*' THEN
20220       IF NOT(wc$(wc_ptr) == search$(sch_ptr)) THEN RETurn 0
20230       sch_ptr = sch_ptr + 1
20240     END IF
20250   END FOR wc_ptr
20260   REMark check for trailing characters in search$
20270   IF sch_ptr = (LEN(search$)+1) THEN RETurn 1 : ELSE RETurn 0
20280 END DEFine Wildcard
```

Figure 2 - Example of program to list only certain types of file

```
100 CLS : CLS #0
110 INPUT #0,'Enter filename wildcard > ';wcard$
120 INPUT #0,'Enter drive name > ';ip$
130 DIR\ram1_tmp,ip$
140 OPEN_IN #3,ram1_tmp
150 INPUT #3,t$ : INPUT #3,t$ : REMark medium details
160 REPeat loop
170   IF EOF(#3) THEN EXIT loop
180   INPUT #3,t$
190   IF Wildcard(wcard$,t$) = 1 THEN PRINT t$
200 END REPeat loop
210 CLOSE #3 : DELETE ram1_tmp
220 STOP
```

Figure 2 shows a simple listing of a program to list only specified files. To list only plain text files, you'd specify *_TXT as the filename wildcard.

To write your own wordsearch program to help with crosswords, etc, suppose we have a wordlist called wordlist_txt, saved as a plain text file on FLP1_.

```
100 CLS : INPUT'Search for ';s$
110 OPEN_IN #3,FLP1_WORDLIST_TXT
120 REPeat loop
130   IF EOF(#3) THEN EXIT loop
140   INPUT #3,t$
150   IF Wildcard(s$,t$) = 1 THEN PRINT
      t$
160 END REPeat loop
170 CLOSE #3
```

How the Routine works

Two pointers are maintained, one running along the characters of the first string containing the wildcard string (wc_ptr), the other along the string being searched (sch_ptr$). If we encounter a '?' in the wildcard, we simply step over the next character in the search string (line 20070).
If we encounter a '*' character in the wildcard, we step along the search string until we find the same character in that string as follows the * in the wildcard (lines 20090 to 20190).
If a non-wildcard character is found, lines 20210 to 20240 increase the pointer while the characters match in both strings, or returns from the routine with value 0 if characters do not correspond.
The routine is case-insensitive, thanks to the use of the QL's "approximately equal" operator ==.

text87 to PC Transfers

by Geoff Wicks

In our news section we report a new version of Norman Dunbar's Stripper program that can convert Quill documents to PDF. Norman's program aroused some interest on the QL-users group with some members asking about the possibility of it being adapted for other QL Word Processors and especially Text87.
This article is about the file formats of QL word processors, and the problems with text87 to PC transfers.
The definitive work on QL file formats is to be found in a document written by Dave Walker. Most of his research was done when he was writing his program Text Tidy. Dave Walker's document is on the QL Today insider documentation CD in the files "textidy" in the folder "psions".
I also did research on file formats for two of my programs, Style-Check and QL-2-PC Transfer. I did not have the benefit of Dave Walker's work when I wrote them and my own voyage of discovery was nowhere near as thorough and detailed as his.
When they were first released people were impressed that my programs automatically detected the file sort. This is much simpler than you might think. Both Quill and Perfection have clear file identifiers. In Quill this is the string "vrm1qdf0" starting at the third byte of the file header. In Perfection the header starts with the identifier. This has two unprintable characters, chr$(255) and chr$(0), followed by "CX". Text87 does not have an identifier, but does have a simple characteristic that the file header starts with the length of the file.

I should perhaps explain what I mean by the file header. Practically all word processors, both QL and PC, start their files with a header giving details such as a file identifier, the length of the text and pointers to various tables containing the document's formatting. This is followed by the text of the document and then usually a lengthy "footer" giving detailed formatting information. Dave Walker has documented in some detail the contents of the Quill "footer".
With my two programs I needed to extract the text from the word processing files. Looking back I did this by what one of my maths teachers described as "brute force and ignorance" methods. With Quill it was easy to find the start of the text section as that was at the end of the header. The end of the file was more difficult, but I discovered that this was marked with chr$(0) and searched for this. I now know from Dave Walker's work that starting at byte 11 of the header is a long word giving the length of the header plus text area.
Perfection was a little more complicated as neither Dave Walker nor I have discovered any information on the file length in the header. However, as with Quill, the start of the text area was easy to find and I took the end of the text section to be the end of the file itself. I was able to get away with this as there are no printable characters after the end of the text section.
Text87 has a header of 72 bytes and starting at byte 13 is a long word giving the length of the text area.

The text sections of Quill and Perfection files are very similar. They consist of the text punctuated by occasional control codes. For example in Quill the control code chr$(15) switches Bold on and off and the code chr$(16) switches underlining on and off. Perfection uses two codes for each of these. Chr$(194) switches bold on and chr$(195) switches it off. Similarly chr$(192) switches underlining on and chr$(193) switches it off.
The text section of Text87 is different. Chr$(27) switches bold on and off and chr$(27) also switches underlining on and off. In other words Text87 does not have separate control codes for each highlighting style. Instead all highlighting and font changes are signalled by chr$(27) and the details of the change are recorded somewhere in the footer. (In fact there is a further complication of which I was unaware until David McCann mentioned it on the QL users group. If you highlight a block of text by making it bold, but then change your mind and remove the highlighting text87 still retains the chr$(27).)
Unfortunately there is no documentation available on the formatting of a text87 file. Even Dave Walker appealed for help on text87 files, and experience suggests that it would be difficult to get the information from the author. The consequence is that QL-2-PC Transfer has only limited functionality with Text87 and does not always transfer highlighting correctly. Similarly we are unlikely to see a Text87 version of Norman Dunbar's Stripper. Text87 is by far the most user unfriendly of the QL word processors, but this is balanced by the fact that it is the most sophisticated, and it has gained the respect of the QL community. There are many users who would like better transfer possibilities to a PC, and anyone who could break the Text87 file formatting code will delight many QL users.
Fortunately all is not lost for Text87 users who own QPC Print. There are several "virtual printer" programs for the PC. One such program is PDF995. This converts practically any printer output to PDF format. It is fully compatible with QPC Print and I have been successful in converting output from several QL programs, including Text87, to PDF.
Another "virtual printer" program is called "Paperless Printer". This produces output in several different formats including HTML, PDF and RTF. Here the results are less promising. HTML appears to work without problems and PDF is successful provided no underlining is present. Unfortunately RTF transfers from QPC Print do not work at all. This is a pity as RTF is the format with the greatest versatility and which most easily permits some editing after the transfer.
QL Today would like to hear from you if you have experience of other "virtual printer" programs, and especially one that could generate either RTF or Word DOC files. Please test it with QPC Print and let us know the results.
One advantage of this approach is that it can also transfer documents containing accented characters as these are converted in the printer drivers.
This brings me conveniently to my final point. I have recently had several requests for the conversion codes between the QL and Windows character sets. These are given in the attached table and should work with most Windows fonts.

Character		QL	PC	Character		QL	PC
pound	£	96	163	u grave	ù	154	249
copyright	©	127	169	u circumflex	û	155	251
a umlaut	ä	128	228	sz ligature	ß	156	223
a tilde	ã	129	227	cent	¢	157	162
a ring	å	130	229	yen	¥	158	165
e acute	é	131	233	acute	´	159	180
o umlaut	ö	132	246	A umlaut	Ä	160	196
o tilde	õ	133	245	A tilde	Ã	161	195
o slash	ø	134	248	A circumflex	Â	162	194
u umlaut	ü	135	252	E acute	É	163	201
c cedilla	ç	136	231	O umlaut	Ö	164	214
n tilde	ñ	137	241	O tilde	Õ	165	213
ae ligature	æ	138	230	O slash	Ø	166	216
oe ligature	œ	139	156	U umlaut	Ü	167	220
a acute	à	140	225	C cedilla	Ç	168	199
a grave	á	141	224	N tilde	Ñ	169	209
a circumflex	â	142	226	AE ligature	Æ	170	198
e umlaut	ë	143	235	OE ligature	Œ	171	140
e grave	è	144	232	micro	µ	176	181
e circumflex	ê	145	234	pi	p	177	112
i umlaut	ï	146	239	iexclamation	¡	179	161
i acute	í	147	237	iquestion	¿	180	191
i grave	ì	148	236	euro	€	181	128
i circumflex	î	149	238	section	§	182	167
o acute	ó	150	243	left arrow	«	184	171
o grave	ò	151	242	right arrow	»	185	187
o circumflex	ô	152	244	degree	°	186	176
u acute	ú	153	250	division	÷	187	247

CONFIG

by David Denham

Configuration is one of those subjects that often tends to get ignored in the hope it will go away. I don't remember seeing much about Config and MenuConfig in print, so here goes.

Configuration is defined as a system designed to allow a program's working settings and startup parameters to be defined and altered as required.

Plain English: Store some settings for a program!

Some information is held in a block within the actual program itself which stores some details that the program needs to know about when starting up.

Here are some examples of options and settings you may get in a configuration block:

- Default drive and/or directory name(s).
- Default printer port device name.
- Sound off or on.
- Program colour settings.
- Language setting for programs which may have English, French, German etc options.
- Mouse speed and response settings.

And anything else the programmer deemed appropriate! In some cases where the setup data is not really appropriate to put into a config block, the programmer may just put one item into a configuration block telling the program where the main data file may be found, e.g. a filename pointer to a printer driver file.

This setup or configuration information is held in a part of the program called a "Configuration Block". This is a system originally devised by Tony Tebby and his team at 1980s QL company QJump. There are two systems, one called Level 1 and a more advanced Level 2. On the QL scene, the term Configuration usually gets abbreviated to Config.

It's all well and good having such a system - you need a program to alter this information when the need arises. It's not a good idea for programs to modify information within themselves - self-modifying code is frowned on for QLs. The two programs which are used to modify configuration settings are:

Config - this is the original program, and is only able to alter level 1 configuration blocks. It does not require pointer environment (although it will use menu extension for file selection if present). Suitable for older systems.

MenuConfig - can alter level 1 and level 2 configuration blocks. Needs pointer environment. Can store and remember settings for individual programs using the level 2 system.

Although most modern QL programs use either Level 1 or Level 2 Config Blocks, some authors have chosen to define their own standards, which usually work well enough, but those programs cannot be altered with Config or MenuConfig. A major advantage of Config Blocks is that they provide a standard system of configuration settings across all programs using the same configuration program.

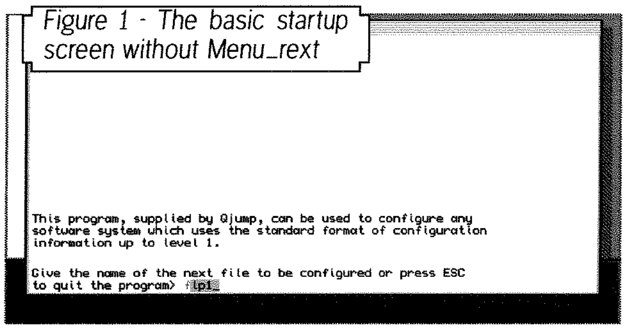
Level 1 CONFIG

This is the most common form you will encounter on older programs. It is very easy to use, but does not have as many facilities as level 2.

The settings can be altered with the original Level 1 Config program, or with the MenuConfig program with one or two limitations.

Config is very easy to use. It asks you for the name of the program to load, loads it, looks at the current settings and asks you a few questions to see how you want to change these settings. Figure 1 is a screen dump of the Config program and you'll see that it can take one of two forms when it starts up, depending on whether the Menu Extension (menu_rext from Jochen Merz) is present. If your system does not have menu_rext installed, it simply asks you to type in the filename

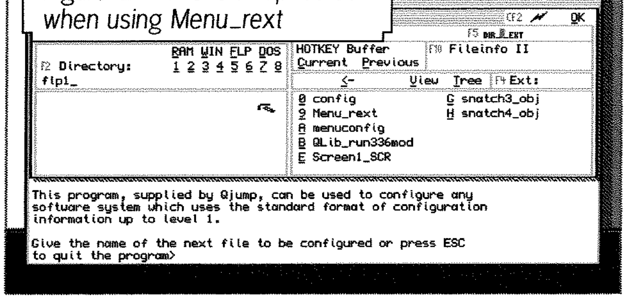
Figure 1 - The basic startup screen without Menu_rext



```
This program, supplied by Qjump, can be used to configure any
software system which uses the standard format of configuration
information up to level 1.

Give the name of the next file to be configured or press ESC
to quit the program> flp1_
```

of the program to be altered. If menu_rext is present, a file selection menu is shown which lets you browse through directories and pick the filename of the program to be altered from a list - see Figure 2. Clearly, if you are used to these menus, the latter is going to be preferable to you, although it does take a little more explanation.

Figure 2 - The startup screen when using Menu_rext



```
This program, supplied by Qjump, can be used to configure any
software system which uses the standard format of configuration
information up to level 1.

Give the name of the next file to be configured or press ESC
to quit the program>
```

The easiest way to specify the filename is to press F3 or click in the box alongside the [F3 Filename:]. This lets you type in the filename of the program to be configured.
You can choose drives by clicking on one of the three letter drive names RAM FLP or MDV (and

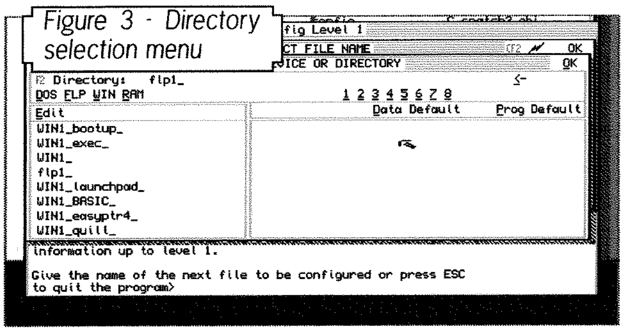


Figure 3 - Directory selection menu

```
fig Level 1
CT FILE NAME                          F2        OK
ICE OR DIRECTORY                                OK
F2 Directory:   flp1_                       <-
DOS FLP WIN RAM               1 2 3 4 5 6 7 8
Edit                          Data Default    Prog Default
WIN1_bootup_
WIN1_exec_
WIN1_
flp1_
WIN1_launchpad_
WIN1_BASIC_
WIN1_easyptr4_
WIN1_quill_
information up to level 1.

Give the name of the next file to be configured or press ESC
to quit the program>
```

whatever other drives exist on your system) and drive number by selecting one of the numbers 1 to 8 under the drive names. Alternatively, left click (Hit) in the box under [F2] and type in a drive and directory name. Or, press F2 or right click (Do) in that box and a directory selection menu appears (see Figure 3) with a list of preset favourite directories on the left and a list of directories you can click on in the right hand box, with the same list of three letter drive names and numbers above them. Once you've finished with that menu, click on OK.
Back in the main menu, you are shown a list of directory names in the left hand window and a list of filenames in the right hand window. Towards the right there is a small box labelled [F4 Ext:] which lets you specify filename extension.
If you left click in this box, you can type in a 3 or 4 letter filename extension for ensuring only programs with that extension get listed, thus shortening the list. For example, if you know that the program you are searching for is Qliberator compiled, it will most probably have the filename extension _obj, so click in the box and type in obj or _obj. If you right click in this box instead, a menu appears with a preset list of extensions - select one of these, or click on the NO option to have all files listed.
If you have a hard disk or romdisq with quite a few directories, you may find yourself a few levels in and want to go back to the root of the drive - use the <- option in the middle of the display to go back from Win1_Quill_ to Win1_ for example.
Once you've specified the filename, Config will proceed to load the program before asking you a series of questions to prompt you to alter settings as required. If the program could not be loaded for some reason, or if no Config data was found within the loaded program, an error message is shown and it goes back to asking you to enter a filename.

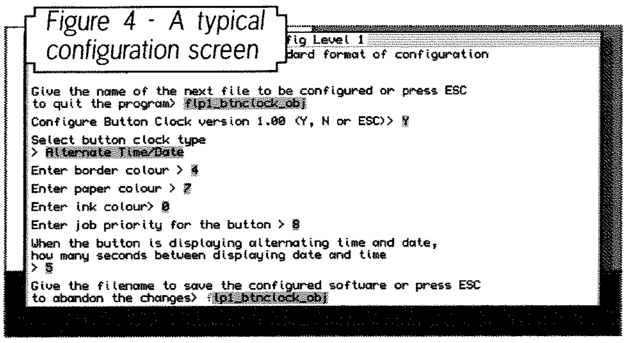


Figure 4 - A typical configuration screen

```
fig Level 1
dard format of configuration

Give the name of the next file to be configured or press ESC
to quit the program> flp1_btnclock_obj
Configure Button Clock version 1.00 (Y, N or ESC)> Y
Select button clock type
> Alternate Time/Date
Enter border colour > 4
Enter paper colour > 7
Enter ink colour> 0
Enter job priority for the button > 8
When the button is displaying alternating time and date,
how many seconds between displaying date and time
> 5
Give the filename to save the configured software or press ESC
to abandon the changes> flp1_btnclock_obj
```

Figure 5 shows a typical configuration screen in progress.
Entries you'll be asked to make fall into just a small number of categories:

1. Type in a string or number and press ENTER. Numbers will usually have a range, for example if asked to specify a weekday number, entry may be limited to a range of 7 numbers such as 0 to 6 or 1 to 7. Strings may be limited in length - drive names may be limited to 5 characters or filenames to 36 characters as dictated by QDOS name limits.
2. Specify a keypress. This is common in programs which allow you to customise keys used to activate menus or commands within the program. Ranges of keypresses may be limited to letters or numbers only, and some "non-standard" keypresses may be represented symbolically, for example, CTRL A may be shown as ^A.
3. Toggled options. This is where a specified set of options only may be available. For example, if Config asks "Sound off or on" the only two options available may be OFF and ON, in which case you are not allowed to type in the words OFF or ON, but should press SPACE to switch between all the available options one at a time.

Some programmers may make use of more advanced selection options to provide more complex facilities, but in general the program's instructions manual should explain these extra options.
With the level 1 config program, you have to go through all the prompts one at a time from start to end. There is no facility to jump direct to, say, setting number three out of seven options.
Once the configuration process is finished, you are asked to specify the filename to save the altered program to. Note that as Config block data is held physically within the program itself, reconfiguring the program really does alter the program code directly - the config block data is held as a small chunk of data embedded somewhere inside the program itself, so there is a small theoretical risk

of damage to the program itself should anything go drastically wrong, such as might happen if there was a power cut halfway through saving the reconfigured program. For this reason, Config lets you specify a new filename for the reconfigured program if you wish to play extra safe. For example, if you loaded FLP1_MYPROG_EXE you may wish to play safe and save the configured copy to FLP2_MYPROG_EXE until you are happy with what you are doing and your confidence grows. In practice, it is unwise to reconfigure your master copy of a program and most of us will be running our programs from a second or backup copy, so this small risk of damage can probably be safely ignored as you will always have another copy to hand, hopefully! The normal saving option will be to save to the same filename as the original, taking the option to overwrite the program with the newly reconfigured version. There is no limit to the number of times you can configure a program unless a programmer has deliberately built in a specified limit, and I know of no such programs. Config will prompt you to press Y or N for Yes or No when it needs to know if it can overwrite the program.

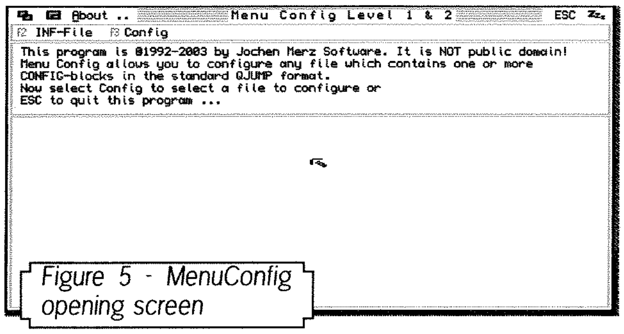
Level 2 CONFIG

Once you have configured a program, you tend not to need to reconfigure it again, unless you obtain an upgrade for a program, in which case it'd be handy for the Config program to be able to remember how you last configured it and just apply the changed settings to the new version of the program rather than have to go through the process of configuring the new version to your preferred settings again. In this way MenuConfig can learn settings from existing programs and update new versions of the program semi-automatically.

This is one of the advantages of the Level 2 version of Config. Usually known as MenuConfig (because it is completely menu driven), it can handle Level 1 config files in pretty much the same sort of way as the original Config program. But because of the different systems used, it cannot store and remember the configuration settings of the older level 1 systems, unfortunately.

For programs using the newer level 2 system, what MenuConfig does is to keep a list of program settings in a file called MenuConfig_INF which it stores on the PROG_USE (programs) default directory. All this is pretty invisible as far as you, the user, is concerned, although it's best to know about it in case anything goes wrong, e.g. if MenuConfig complains it can't find its settings file, the chances are that PROG_USE is pointing to the wrong directory.

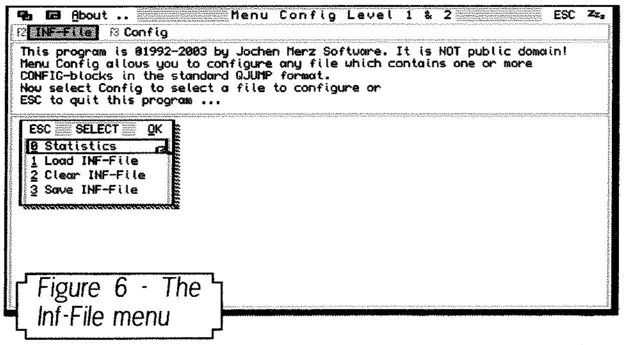
Jochen Merz Software has allocated values called "Config IDs" to software publishers, which are unique to each publisher. This means that between the publisher ID and each program's own ID it is extremely unlikely if not impossible for MenuConfig to confuse two programs, so it can safely store settings for all programs using level 2 configuration information.



```
This program is ©1992-2003 by Jochen Merz Software. It is NOT public domain!
Menu Config allows you to configure any file which contains one or more
CONFIG-blocks in the standard QJUMP format.
Now select Config to select a file to configure or
ESC to quit this program ...
```

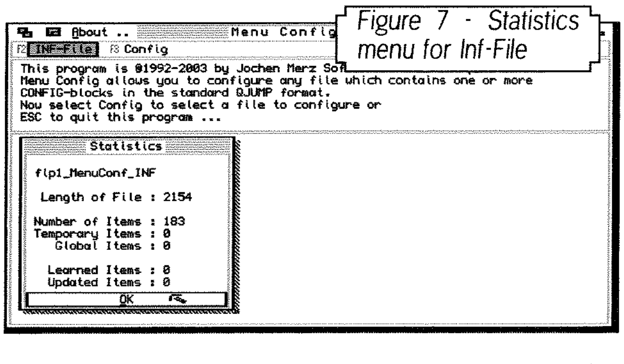
Figure 5 - MenuConfig opening screen

Figure 5 shows the opening screen of MenuConfig. The two main facilities are [F2 INF-File] and [F3 Config]. Selecting the first brings up the little INF-File menu from where you can see details of how many entries are stored in the MenuConfig_INF file, load a specified MenuConfig_INF file from a specified location, save to a specified location (e.g. to make a backup copy of MenuConfig_inf), or even clear it out completely if you want to restart completely from scratch.



```
This program is ©1992-2003 by Jochen Merz Software. It is NOT public domain!
Menu Config allows you to configure any file which contains one or more
CONFIG-blocks in the standard QJUMP format.
Now select Config to select a file to configure or
ESC to quit this program ...
```

Figure 6 - The Inf-File menu

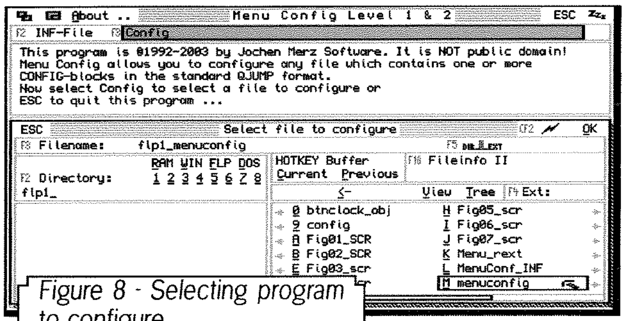


```
This program is ©1992-2003 by Jochen Merz Sof
Menu Config allows you to configure any file which contains one or more
CONFIG-blocks in the standard QJUMP format.
Now select Config to select a file to configure or
ESC to quit this program ...
```

```
flp1_MenuConf_INF

 Length of File : 2154

Number of Items : 183
Temporary Items : 0
   Global Items : 0

  Learned Items : 0
  Updated Items : 0
```

Figure 7 - Statistics menu for Inf-File

Figure 6 shows the options available in the INF-File menu, and figure 7 shows the information listed in the Statistics menu, such as number of items stored.



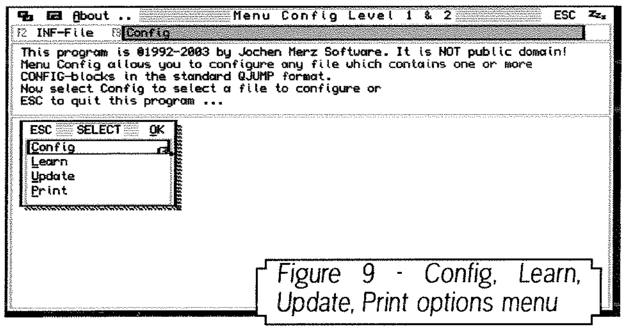
```
This program is ©1992-2003 by Jochen Merz Software. It is NOT public domain!
Menu Config allows you to configure any file which contains one or more
CONFIG-blocks in the standard QJUMP format.
Now select Config to select a file to configure or
ESC to quit this program ...
```

```
F3 Filename:    flp1_menuconfig
F2 Directory:
flp1_

0 btnclock_obj      H Fig05_scr
9 config            I Fig06_scr
A Fig01_SCR         J Fig07_scr
B Fig02_SCR         K Menu_rext
E Fig03_scr         L MenuConf_INF
                    M menuconfig
```

Figure 8 - Selecting program to configure

Figure 8 shows how you select the program to configure, using a similar file selection menu to that used by level 1 Config. MenuConfig can configure itself - just tell it to load itself to practice on!

Once selected, the program is loaded and a small selection menu is displayed (see Figure 9).



```
This program is ©1992-2003 by Jochen Merz Software. It is NOT public domain!
Menu Config allows you to configure any file which contains one or more
CONFIG-blocks in the standard QJUMP format.
Now select Config to select a file to configure or
ESC to quit this program ...
```

Figure 9 - Config, Learn, Update, Print options menu

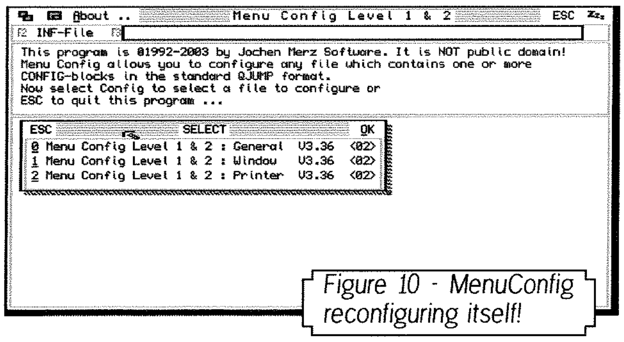
This has 4 options - Config, Learn, Update and Print.

Learn - adds the settings of this program to the list in MenuConfig_INF and then return to main menu. In other words, Learn the settings of the program you just loaded.

Update - look in the list of items stored in MenuConfig and if the program's details are already stored, apply them to the program you just loaded. This is the option you may use when updating to a new version of an existing program.

Print - as you might expect, prints a list of settings to a specified destination such as SER1 or PAR. Output can also be to a file if preferred.

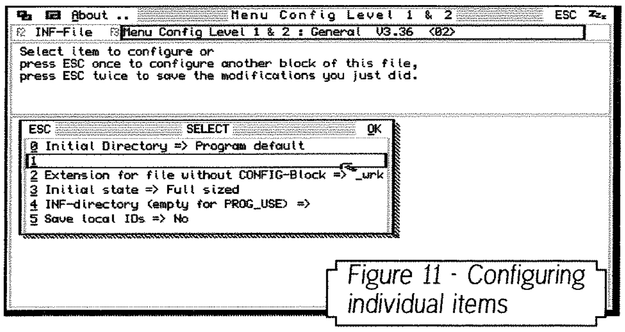
Config - this is the normal option of course. It brings up a menu containing a list of config block sections (programs can have more than one Config block if the programmer so deems).



```
This program is ©1992-2003 by Jochen Merz Software. It is NOT public domain!
Menu Config allows you to configure any file which contains one or more
CONFIG-blocks in the standard QJUMP format.
Now select Config to select a file to configure or
ESC to quit this program ...

0 Menu Config Level 1 & 2 : General  V3.36  <02>
1 Menu Config Level 1 & 2 : Window   V3.36  <02>
2 Menu Config Level 1 & 2 : Printer  V3.36  <02>
```

Figure 10 - MenuConfig reconfiguring itself!

Figure 10 shows an example of MenuConfig configuring a copy of itself, and you'll see three sections displayed, in this case, General, Window and Printer. Click on any of these and it will show a list of individual items within that section of the

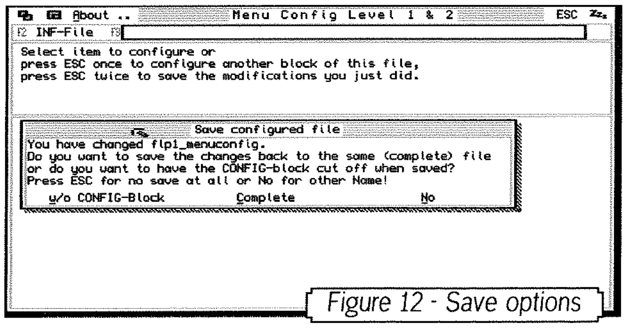


```
Menu Config Level 1 & 2 : General  V3.36  <02>

Select item to configure or
press ESC once to configure another block of this file,
press ESC twice to save the modifications you just did.

0 Initial Directory => Program default
1
2 Extension for file without CONFIG-Block => _wrk
3 Initial state => Full sized
4 INF-directory (empty for PROG_USE) =>
5 Save local IDs => No
```

Figure 11 - Configuring individual items

config block (see Figure 11). In other words, related items in a config block can be grouped together. And a useful difference between level 2 and level 1 programs is that you can choose which item to alter at any time, rather than having to go through all items from first to last in order.

Click on the item to be altered and a window is displayed asking for you to enter the required information. This may ask you to enter a string or a value, or a selection menu may be displayed asking you to select one of a short list of items. Once you've finished with a given set of options, click on the OK icon where present to go back to the previous menu or ESC to quit.



```
Select item to configure or
press ESC once to configure another block of this file,
press ESC twice to save the modifications you just did.

You have changed flp1_menuconfig.
Do you want to save the changes back to the same (complete) file
or do you want to have the CONFIG-block cut off when saved?
Press ESC for no save at all or No for other Name!
```

Figure 12 - Save options

If changes have been made to the program, it will ask if you wish to save the altered program (Figure 12). It offers the option of saving with or without the config block, though I can't imagine why you would want to save the reconfigured program without the config block unless it's to ensure that the program minus config block is smaller in size or some such reason.

There is also an option to save under a different filename to the original, somewhat confusingly called NO. I presume it's NO to the option to save back to the same file! If you select NO it brings up a filename entry box for you to enter a different filename for the program to be saved under.

Once you have saved the reconfigured program, you may notice a flashing picture of an arrow pointing to a disk in the top right corner of the

display. This tells us that something needs saving - click on ESC to quit and you will be prompted to save an updated copy of the MenuConfig_INF file. Most of the time the reply will be Y for Yes, but you do have the option not to update it if you prefer for some reason.
That's it really. Configuration is one of those subjects that seems intimidating at first, but which you quickly get used to. Most modern programs, especially pointer driven ones, have configuration blocks built in and MenuConfig makes the task of reconfiguring modern QL programs which may have extensive configuration block settings much easier than the rather rigid system used by the old level 1 system. And remember that Level 2 MenuConfig is also able to alter settings in older Level 1 systems.

XMON and XTV

by Dilwyn Jones

In his Start Here article in the April-June 2006 issue, Roy Wood describes how the first screen appears as a small box somewhere on the screen when you start up a system in a high resolution mode.
This inspired me to dig out a couple of old routines I use to add a couple of procedures called XMON and XTV. These create windows like those created by the WMON and WTV commands, but make use of the whole screen. Window channels #2, #1 and #0 now make use of the whole screen.
The procedures make use of the SCR_XLIM and SCR_YLIM functions to return screen size details so that the windows can be scaled to fit whatever screen resolution you use.
An OUTLN command is included in both procedures to ensure windows are correctly outlined (to ensure menu extension etc works properly).
Entering the command XMON sets monitor-style windows (i.e. window channels #1 and #2 side by side occupying half screen width each) and XTV sets TV-style windows (i.e. window channels #1 and #2 overlap, although unlike WTV the windows occupy the full width of the screen as it's assumed if you are using high resolution you won't be using a TV and won't suffer from edge cutoff).

```
30000 DEFine PROCedure Xtv
30010   OUTLN #0,SCR_XLIM,SCR_YLIM,0,0
30020   WINDOW SCR_XLIM,SCR_YLIM-52,0,0 : WINDOW #2,SCR_XLIM,SCR_YLIM-52,0,0 :
        WINDOW #0,SCR_XLIM,52,0,SCR_YLIM-52
30030   BORDER #1,1,255 : BORDER #2,1,255 : BORDER #0,1,255
30040   CLS : CLS #2 : CLS #0
30050 END DEFine Xtv
30060 :
30070 DEFine PROCedure Xmon
30080   OUTLN #0,SCR_XLIM,SCR_YLIM,0,0
30090   WINDOW SCR_XLIM/2,SCR_YLIM-54,SCR_XLIM/2,0 : WINDOW #2,SCR_XLIM/2,
        SCR_YLIM-54,0,0 : WINDOW #0,SCR_XLIM,52,0,SCR_YLIM-52
30100   BORDER #1,1,255 : BORDER #2,1,255 : BORDER #0,1,255
30110   CLS : CLS #2 : CLS #0
30120 END DEFine Xmon
```

WMON and WTV

These commands can do more than just reset SBASIC's windows to the startup sizes in the top left corner of the high resolution screen. So many old programs just sit in the top left of the screen, leaving a lot of redundant space to the right and below. So the SBASIC WMON and WTV commands can take extra parameters to specify where the top left corner of SBASIC's windows will fall. There are two versions of each command:

```
WMON mode_number, x and y co-ordinate

WMON mode_number, x co-ordinate,
   y co-ordinate
```

The co-ordinates are given in pixels across and down the screen. If only one co-ordinate is given, the command assumes that both are the same:

```
WMON 4,100
```

This command sets monitor windows in 4 colour screen mode, with the top left corner 100 pixels across the screen and 100 pixels down.

```
WMON 8,50,100
```
This command sets monitor windows in 8 colour screen mode, with the top left corner 50 pixels across and 100 pixels down. Replace WMON with WTV for TV-style windows.

If you want to place the basic windows at the bottom right of the screen to allow older fixed location programs the use of the top left of the screen, try using SCR_XLIM and SCR_YLIM to work out where to put the windows. This assumes that the SBASIC windows are to use a 512x256 pixel area just like the standard QL screen:

```
WMON 4,SCR_XLIM-512,SCR_YLIM-256
```
The mode number can be left out unless you specifically want to set a given mode. In fact, it's often better not to specify it since window contents can be preserved. Simply leave the parameter before the first comma blank:

```
WMON ,SCR_XLIM-512,SCR_YLIM-256
```

A useful possibility is to put these commands onto a hotkey as a command, using HOT_CMD:

```
ERT   HOT_CMD("w","wmon   ,scr_xlim-512,
scr_ylim-256")
```

Please note that this article is only applicable to SMSQ/E systems, since QDOS does not support high resolution screens.

DIY Hardware Add-ons for your Sinclair Computer - Part II

by Phoebus R. Dokos, B.Sc

B. Project 3

DIFFICULTY LEVEL: (Easy/Intermediate)

ZXCF Compact Flash Card reader and NMI/Reset buttons for the ZX Spectrum

IMPORTANT:
THIS PROJECT INVOLVES SOLDERING DIRECTLY ONTO THE Z80 CPU OF THE SPECTRUM AS WELL AS POTENTIALLY CUTTING BOTH THE ADAPTER AND SPECTRUM CASE. IF YOU DO NOT POSSESS A LOW POWER IRON AND SUFFICIENT SOLDERING SKILLS, DO NOT ATTEMPT IT WITHOUT QUALIFIED HELP!!!!

Introduction

All ZX Spectrum models with the exception of the +3 and +2A (Or any Spectrum with a disk drive) have one annoying thing in commonà Extremely slow program loading. Although superior sound technology and improved Speed Loader techniques have boosted Spectrum Speed Loaders to their limit of 427% of the standard loading speed, it still takes an average 1 min. or more to load a given 48K program (not to mention the 128K ones!) - Remember that speed loaders themselves have to be loaded in memory first at a standard speed before they can work! Moreover, you need to haul a sound source (these days a CD or even a full blown PC), the cables create a mess and the use of HDDs or FDDs adds up to the bulkiness of the package.
The ZXCF by Sami Veehma, solves these problems, by providing, not only a CF card interface that provides ample room for even the largest multi-part programs of the Spectrum, but also up to 1 Mbyte of on board static RAM as well as the ability to have a task switcher, restore on-boot to a previously saved position, memory snapshot as well as effortless switching of ROM versions and compatibility with IF2 ROMs.
All these come at a cost - You won't be able to use most peripherals with a Shadow ROM that use the /ROMCS line (such as the IF1 - but not the ZX Printer or the T/S2048 - Alphacom 32 printer which work fine) but the benefits are enormous and the Spectrum becomes one powerful and really portable machine.

You can get the ZXCF adapter directly from Sami Veehma's website (if he has one available) here: **http://user.tninet.se/~vjz762w/** or build it yourself, using the provided schematics. Mind you this is not a job for the faint hearted as it involves surface mounted components!!!!

TF Services

Compswitch

A UK 4-way trailing socket designed to switch off computer peripherals automatically when the computer is switched off, or (in the case of an ATX computer) when it auto-powers down. *Compswitch* has one control socket, and three switched sockets.
Can be used with lights/hifi/monitors—ie a QL monitor can be used as a switch control.

Cost £24

superHermes

A major hardware upgrade for the QL
All Hermes features (working ser1/2 at 19200, independent baud rates/de-bounced keyboard/ keyclick) IBM AT kbd I/F // HIGH SPEED RS232 at 57600// serial mouse port and 2 other RS232 inputs// 3 I/O lines // EEPROM

Cost (including manual/software)	**£90** (£92/£93)
IBM AT UK layout Keyboard	**£11** (£13/£15)
Serial mouse	**£8** (£8.50/£9)
Capslock/scrollock LED	**£1** (£1.50/£1.50)
Keyboard or mouse lead	**£3** (£3.50/£3.50)
High speed serial (ser3) lead	**£4** (£4.50/£4.50)

Hermes available for £25 (£26/£27) Working ser1/2 and independent input, debounced keyboard.

SuperHermes LITE: All Hermes features (see above) + an IBM AT keyboard interface only.
Cost (incl keyboard lead) .... **£53** (£54/£55)

QL REPAIRS (UK only)

Fixed price for unmodified QLs, excl microdrives. QLs tested with Thorn-EMI rig and ROM software.

£27 incl 6 month guarantee

Minerva

The ORIGINAL system operating system upgrade

OTHER FEATURES COMMON TO ALL VERSIONS

DEBUGGED operating system/ autoboot on reset of power failure/ Multiple Basic/ faster scheduler- graphics (within 10% of lightning) - string handling/ WHEN ERROR/ 2nd screen/ TRACE/ non-English keyboard drivers/ "warm" fast reset. V1.97 with split OUTPUT baud rates (+ Hermes) & built in Multibasic.

First upgrade free. Otherwise send £3 (+£5 for manual if requd). Send disk plus SAE or two IRCs

MKI...**£40** (£41/£43) MKII...**£65** (£66/£67)

MINERVA RTC (MKII) + battery for 256 bytes ram. CRASHPROOF clock & I^2C bus for interfacing. Can autoboot from battery backed ram. Quick start-up.

QL RomDisq

Up to 8 mbyte of flash memory for the QL
A small plug in circuit for the QL's ROM port (or Aurora) giving 2, 4 or 8 mbytes of permanent storage - it can be thought of as a portable hard disk on a card, and reads at some 2 mbytes per second.
Think of it - you could fully boot an expanded QL, including all drivers/SMSQ etc off **RomDisq** at hard disk speed with only a memory expansion needed.

2 mbytes RomDisq	**£39** (£40/£41)
4mbytes RomDisq	**£65**(£66/£67)
8 mbytes RomDisq	**£98** (£99/£100)
Aurora adaptor	**£3** (£3.50/£4)

MPLANE

A low profile powered backplane with ROM port

A three expansion backplane with ROM port included for RomDisq etc. Aurora can be fitted in notebook case and powered off single 5V rail - contact QBranch for details. Two boards (eg Aurora and Gold Card/Super Gold Card/Goldfire fixed to base. Suitable for Aurora (ROM accessible from outside) & QL motherboard in tower case. Specify ROM facing IN towards boards, or OUT towards back of case.

Cost .... **£34** (£35/£36)

I2C INTERFACES

Connects to Minerva MKII and any Philips I^2C bus

Power Driver Interface 16 I/O lines with 12 of these used to control 8 current carrying outputs (source and sink capable)

2 amp (for 8 relays, small motors)	**£40** (£43/£44)
4 amp total (for motors etc)	**£45** (£48/£50)

Relays (8 3a 12v 2-way mains relays (needs 2a power driver) .... **£25** (£28/£29)

Parallel Interface Gives 16 input/output lines. Can be used wherever logic signals are required.......... **£25** (£27/£28)

Analogue Interface Gives eight 8 bit analogue to digital inputs (ADC) and two 8 bit digital to analogue outputs (DAC). Used for temp measurements, sound sampling (to 5 KHz), x/y plotting.................. £30 (£31/£32)

Temp probe (-40°C to +125°C)	**£10** (£10.50/£11)
Connector for four temp probes	**£10** (£10.50/£11)
Data sheets	**£2** (£2.50/£3)
Control software & manual (for all I/F)	**£2** (£2.50/£3)

QL SPARES

Keyboard membrane	**no longer on sale**
1377 PAL	**£3** (£3.50/£4)
Circuit diagrams	**£3** (£3.50/£4)
68008 cpu or 8049 IPC	**£8** (£8.50/£9)
8301/8302 or JM ROM or serial lead	**£10** (£10.50/£11)
Power supply (sea mail overseas)	**£12** (£19/£23)

Other components (sockets etc) also available

Prices include postage and packing (Airmail where applicable) Prices are: UK (Europe /Rest of world). Payment by cheque drawn on bank with UK address,/postal order or CASH! I can no longer accept card payments as UK only does PDQ transaction. SAE or IRC for full list and details 22 Feb 03

29 Longfield Road, TRING, Herts, HP23 4DG
Tel: +44 (0) 1442-828254 Fax/BBS: +44 (0) 1442-828255
tony@firshman.co.uk http://www.firshman.co.uk

We will look at two methods of installation. One internal (Recommended for the Spectrum+ and above but not for the 16K or 48K unless you have specialized tools) and one external (Works fine with everything and can also work with IF1 with some software fiddling).

Tools Needed

- Small head Philips Screwdriver
- IC Extractor and/or Small head Flat Screwdriver
- Anti-static IC carrier or Aluminium covered polystyrene
- Small Angled Cutter
- Cable stripper capable of stripping 20 AWG wire and above
- Low power soldering iron (such as Antex 12W or similar) and solder
- De-soldering braid and/or de-soldering pump
- Nibbler (to be able to cut the side of the Spectrum for a nice professional-looking CF slot)
- High grade file (for smoothing of jagged plastic etc)
- Small drill for reset and NMI button holes
- Dremel type hobbyist drill for modification of the ZXCF PCB if needed
- Thick double-sided adhesive tape
- IDC Crimper or desktop vise (For the IDC connector)
- Small hacksaw (for external installation if larger Card edge connector than 56 conductor is used)
- Small pen knife or scissors to separate the wires in the flat cable

Parts Needed

a. **Internal Installation**

- ZXCF adapter!
- 34 conductor, flat cable around 20 AWG thickness
- 2 x SPST momentary switches (for reset and NMI or just one for NMI if your computer already has a reset switch)
- 33nF Capacitor (Ceramic)
- 1MΩ Resistor
- 1 On/Off mini toggle switch (for displacement of ZXCF's "Upload" Jumper
- 34 pin IDC crimp female connector
- Length of 2 wire cable for NMI & Reset buttons
- 2-3 cm (approx. 1") of 5 conductor flat cable (0.56" centre displacement) for extension of the keyboard connector
- Software (ResiDOS 1.84 and above)

b. **External Installation**

- ZXCF adapter!
- 34 conductor, flat cable around 20 AWG thickness
- 2 x SPST momentary switches (for reset and NMI or just one for NMI if your computer already has a reset switch)
- 33nF Capacitor (Ceramic)
- 1MΩ Resistor
- 1 On/Off mini toggle switch (for displacement of ZXCF's "Upload" Jumper
- 34 pin IDC crimp female connector
- 56 pin or larger (ie. ISA-16) Card Edge Connector 0.100" centre – Wire wrap is recommended - or in case a breadboard is used, the solder through version
- Length of 2 wire cable for NMI & Reset buttons
- Electronics Project Box to house the assembly
- Software (ResiDOS 1.84 and above)

Preparation

Depending on the installation type, you could use shorter (external) or longer (internal) cables, however you should keep in mind that the maximum length should be kept under 20cm – a total length of 10 cm or less would be ideal.

To begin the installation in any case you should prepare the 34 conductor flat cable & IDC connector (to be plugged in the ZXCF). Using an IDC crimper or a small desktop vise, crimp the connector to the cable taking care to align pin 1 of the IDC connector to align with the red line on the flat cable in such a way that when the case is closed the wire comes from the bottom side (pin 2 side) of the connector (figure 4).

Next fray the wires one by one so that the form of the flat cable is only retained inside the IDC connector.

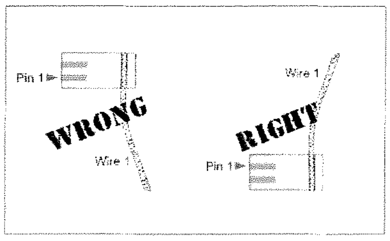


Figure 4 -IDC & Cable correct placement

According to which model of the ZX Spectrum you are going to be installing the ZXCF and how you will also have to do the following (Table 1):

	Machine	
Wire No.	16K/48K/+/128/+2	+2A/+3
27 (/ROMCS)	Allow for extra length (7cm)	Cut (Not Used)
29 (/ROM10)	Cut (Not Used)	Allow for extra length (7 cm)
31 (/ROM20)	Cut (Not Used)	Allow for extra length (7 cm)

Table 1 - Internal Installation Wire Selection

You will only need to allow for extra length on an internal installation.

Internal Installation

DIFFICULTY LEVEL: Intermediate

Next, strip a small length at the end of each wire (approx 1-2 mm) and "solder dip" it – this procedure will improve the speed of soldering onto the CPU thus minimising the potential damage to it.

Finally we will need to prepare the ZXCF board itself if it is to be installed internally in a 16K/48K (not +) Spectrum, by drilling a hole on the right edge of the CF connector (seeing it from the component side) (figure 5).

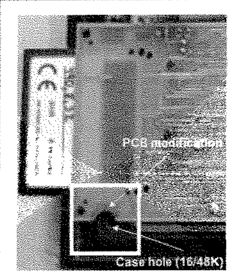


Figure 5 - ZXCF PCB modification

Once this is done, it is time to open up our Spectrum. Refer to the instructions on projects 1&2 on how this is done for models up to 128K. For +2 and above, please refer to the technical reference available at the World of Spectrum super site **http://www.worldofspectrum.org/**
Additionally remove the heat sink from the LM7805 voltage regulator. This has either the form of a long aluminium plate that is located right above the expansion port or for older ZX Spectrums (PCB Issues 1 through 5) a weird inverted gothic "S" shape.
To do so, use pliers and a flat screwdriver. Make sure that you maintain the insulating washer as well as the screw and nut. These are very important as a longer screw or thicker nut will prevent the PCB to sit properly upon re-assembly and can damage your PCB. It is also a good idea now, to remove the ULA and place it on the anti-static IC carrier taking care as not to bend its pins.

Once the PCB is in front of you, the first order of business is to de-solder and remove the leftmost keyboard connector and place it on a little extension. This is needed as the best location for the CF adapter sits right above it thus making it impossible to have both the CF adapter and the left keyboard tail in at the same time. The extension solves that problem.

***CAUTION:** Take care of recording the orientation of the keyboard connector. It MUST face the opposite direction than the right (larger) one. It is very easy to be confused by it and have to do it all over again after an arduous soldering session on the CPU.*

Take the 5 conductor flat cable and put it in the holes left by the removed connector, then solder the other end to the connector. Ta-dah! You now have a nifty keyboard extension (Mind you that you can do that to the QL and ZX-81 if for some reason your membrane is fine but the tails are broken!)

Once this is done, arm yourself with a hot iron, and the following table and two illustrations (Table 2 and figures 6 and 7). If you have done the solder dipping correctly, you will need no solder at this stage as you can use the solder already on the wires.

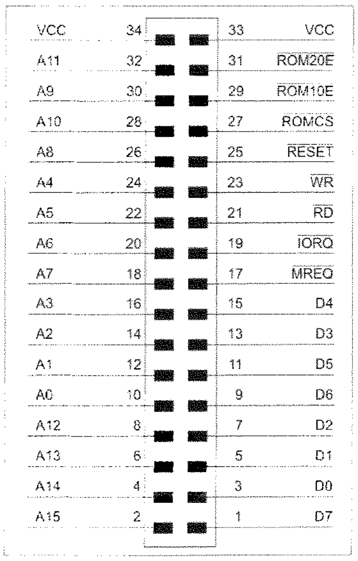


Figure 6 - ZXEXP BUS Pinouts

Take each wire from the IDC connector; starting with wire 1 (marked by a red line throughout) and using the table solder it on the appropriate pin of the Z80 processor.

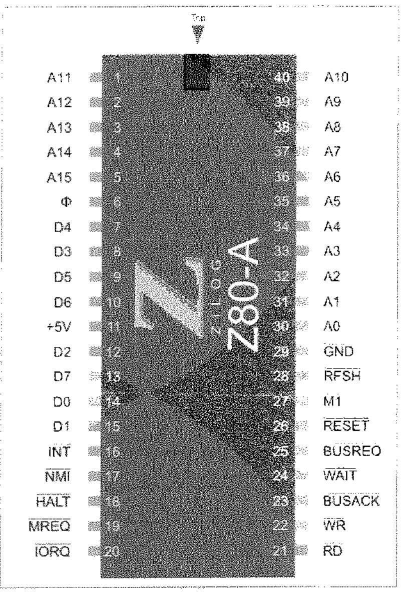


Figure 7 - Z80 CPU Pinouts

***WARNING:** Different Spectrums have different Z80 orientations. Use figure 8 and the orienting notch on the chip to figure out the correct number of each pin.*

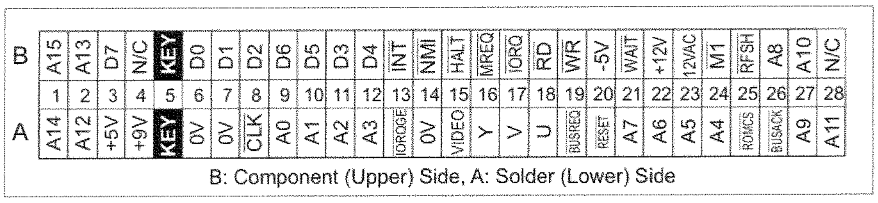
***NOTE:** The CPU is a 40 pin chip usually at the right end of the mainboard for the 16/48/+ and 128K. For the location of the CPU of the Amstrad made models, consult WOS (see URL above). Pin numbering as seen on figure 7 is starts on the top left side of the IC, and runs anti-clockwise around it. That makes for the Z80 CPU (40 pin DIP package) pin 1 the upper leftmost, pin 20 the lower leftmost, pin 21 the lower rightmost and pin 40 the upper rightmost.*

The connector used by ZXCF conforms to a custom bus devised by ZXCF's creator Sami Veehma, called ZXEXP.It may seem odd at first but one look at the Spectrum's expansion connector will provide insight on why it is so.

Signal Name	ZXEXP Pin No.	Z80 Pin No.	Notes
D7	1	13	
A15	2	5	
D0	3	14	
A14	4	4	
D1	5	15	
A13	6	3	
D2	7	12	
A12	8	2	
D6	9	10	
A0	10	30	
D5	11	9	
A1	12	31	
D3	13	8	
A2	14	32	
D4	15	7	
A3	16	33	
/MREQ	17	19	
A7	18	37	
/IORQ	19	20	
A6	20	36	
/RD	21	21	
A5	22	35	
/WR	23	22	
A4	24	34	
/RESET	25	26	
A8	26	38	
/ROMCS	27	NC	Connects on Expansion Port on 16/48/128/+2. Not Connected on +2A/+3
A10	28	40	
/ROM10	29	NC	Connected on Expansion Port on +2A/+3. Not connected on 16/48/128/+2
A9	30	39	
/ROM20	31	NC	Connected on Expansion Port +2A/+3. Not connected on 16/48/128/+2
A11	32	1	
VCC	33	11	
GND	34	29	

Table 2 - ZXEXP - Z80 Pin equivalence

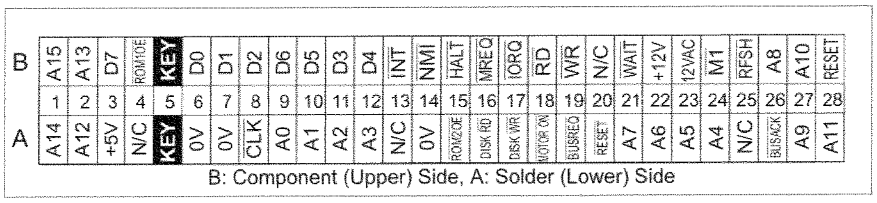
After this is done, and depending on the model, either connect wire 27 to A-25 of the edge connector (figure 8) – making sure there's ample room for a peripheral to be plugged in – or wire 29 to B-4 and wire 31 to A-15 (figure 9)



B	A15	A13	D7	N/C	KEY	D0	D1	D2	D6	D5	D3	D4	INT	NMI	HALT	MREQ	IORQ	RD	WR	-5V	WAIT	+12V	12VAC	M1	RFSH	A8	A10	N/C
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
A	A14	A12	+5V	+9V	KEY	0V	0V	CLK	A0	A1	A2	A3	IORQGE	0V	VIDEO	Y	V	U	BUSREQ	RESET	A7	A6	A5	A4	ROMCS	BUSACK	A9	A11

B: Component (Upper) Side, A: Solder (Lower) Side

Figure 8 - ZX Spectrum 16K/48K/+/128 Edge Connector



B	A15	A13	D7	ROM1OE	KEY	D0	D1	D2	D6	D5	D3	D4	INT	NMI	HALT	MREQ	IORQ	RD	WR	N/C	WAIT	+12V	12VAC	M1	RFSH	A8	A10	RESET
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
A	A14	A12	+5V	N/C	KEY	0V	0V	CLK	A0	A1	A2	A3	N/C	0V	ROM2OE	DISK RD	DISK WR	MOTOR ON	BUSREQ	RESET	A7	A6	A5	A4	N/C	BUSACK	A9	A11

B: Component (Upper) Side, A: Solder (Lower) Side

Figure 9 - ZX Spectrum +2A/+3/+3A Edge Connector

Now it will be time to add a reset button (if on a ZX Spectrum 16K/48K). The best way for this is to locate the capacitor labeled C27 and solder a wire on each pole. Then solder the tails to one of the momentary switches you have.

Next you should add the NMI button, which is crucial for the operation of the ZXCF (Task switching among other things depends on a NMI (Non-Maskable Interrupt) being issued to the CPU).

The NMI switch is a little trickier than the Reset as it requires some minimal "debouncing". This is because most momentary switches have bounce (that being the continuous open-close of the switch until the contacts settle mechanically). The NMI is negative triggered on the Z80 and it requires a minimum pulse duration of 80 nsec.
To achieve the desired debouncing without an unstable multi-vibrator (555) based circuit - after all we aim for simplicity here- you will need an RC circuit right after the GND and before the switch to the NMI pin of the CPU. Follow the instructions on figure 10 for proper wiring. If you are installing on a rubber keyed Spectrum, try to keep the tails on the components to a minimum length. For any other Spectrum, there's ample room to install them left or right of the mainboard. On my installation to the Spectrum+ I added all switches to the right side (NMI being on the hardest to hit accidentally bottom side).

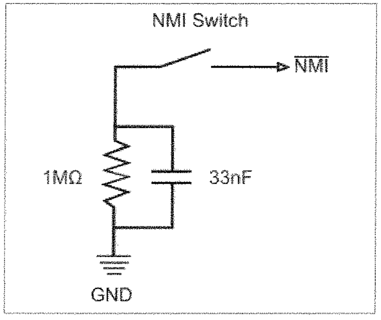


Figure 10 - Debounced NMI button diagram

Finally we need to solder the upload switch to cables that will attach themselves to the upload jumper on the ZXCF. I would recommend using a LED cable from an old PC as it already has the female connectors on one edge and it will save you a lot of trouble trying to either soldering it on it - or - trying to crimp it yourselves.

Once this is done as well, it is time to prepare the case for the CF slot and the switches' holes.
The best place to do that on a ZX Spectrum 16/48/+ is on the lower left side of the case (The CF card will have to be inserted from the left). Cut a hole 0.75 cm by 4.6 cm (approx 0.3" by 1.8") with the nibbler and make sure it looks neat! Then, find a proper space for the Reset (for 16/48), NMI and Upload switches. That should be on the upper side (next to the modulator) on a rubber keyed Spectrum. The best place for a + is on the empty right side. As mentioned above I put my NMI on the bottom and my upload switch on the right on my Spectrum+ (Figure 11)

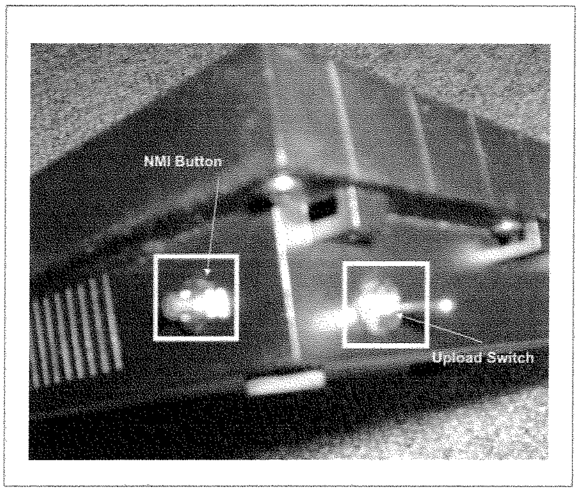


Figure 11 - ZX Spectrum+ Lower right side with NMI and Upload Switches

If you own a 128K, it gets a little trickier. The best thing to do is use a ZXREG fix from Sami Veehma and completely remove the heat-sink on the right side. You will have room for both switches AND the ZXCF there (Figure 12).

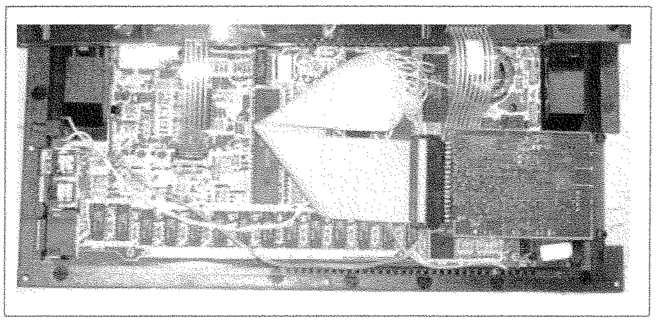
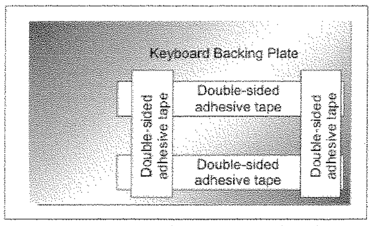


Figure 12 - ZX Spectrum 128 K with ZXCF, ZXREG installed and heatsink removed

The Amstrad models have a lot of room, so finding an appropriate place won't be as difficult. Use your best judgment.

Now it will be time to put everything together. I have found that the best way to secure the ZXCF with minimum intrusion is to stick it on the top cover under the membrane plate with double tape. Because a screw is getting in the way, use double tape in the manner displayed in figure 13 and stick it with the ZXCF's component side facing the Spectrum's PCB.


Keyboard Backing Plate
Double-sided adhesive tape
Double-sided adhesive tape
Double-sided adhesive tape
Double-sided adhesive tape
Figure 13 - Proposed Double Sided Tape Mounting

WARNING: If you have a 16/48K Spectrum, make sure that the hole you have precut on the ZXCF lines up with the Spectrum's case holes as it won't be easy to remove the ZXCF without some effort if that doesn't happen.

Next replace the heat sink with care not to bend the cables of the ZXEXP bus excessively and put the board back into place making sure to route the cable in such a way that it is not in the way of a case hole or in danger of breaking.
Once this is done, put the switches in their respective holes and mount them with their hardware. Next, plug the ZXEXP 34 IDC connector to the ZXCF and put the keyboard tails back in their sockets.
Close the cover carefully (without forgetting the legs on a +/128K!) and screw everything back together (Figure 16).
If your machine starts (!), you will then, be ready to install the software.

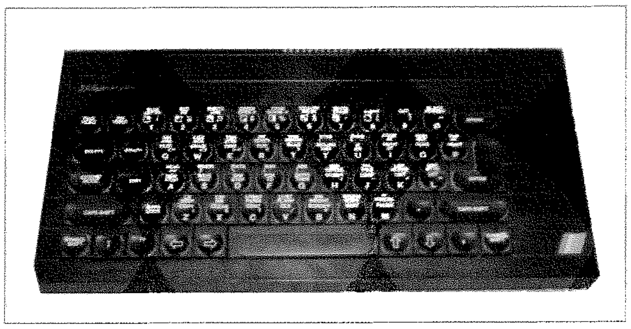


Figure 16 - Can you tell if this Spectrum has a Compact Flash Card and 512K RAM?

External Installation

DIFFICULTY LEVEL: EASY

This is MUCH easier than the internal installation, but not nearly as neat!
First you will need to prepare the Card Edge Connector if it isn't a 56 way one. You do that by using a hacksaw and cutting it to size.
Next locate and solder (or if your electronics store provided you with one slide it in) the upper and lower contacts of the key location – that would be location 5 (start to count from 1).
Finally using the figures of the edge connectors and the following table (Table 3) solder the ZXEXP connector to the bus. A = Solder side, B = Component side on the ZX Spectrum mainboard, ZXEXP = Bus for ZXCF)

ZX Connector	Signal Name	Connect to:	Notes
A1	A14	ZXEXP4	
A2	A12	ZXEXP 8	
A3	+5V	ZXEXP 33	
A6	GND	ZXEXP 34	
A7	GND	RESET BUTTON	
A9	A0	ZXEXP 10	
A10	A1	ZXEXP 12	
A11	A2	ZXEXP 14	
A12	A3	ZXEXP 16	
A14	GND	NMI BUTTON	DEBOUNCE!!!
A15	ROM20	ZXEXP 31	+3 / +2A – NC on ALL others
A20	RESET	ZXEXP 25 & RESET BUTTON	
A21	A7	ZXEXP 18	
A22	A6	ZXEXP 20	
A23	A5	ZXEXP 22	
A24	A4	ZXEXP 24	
A25	ROMCS	ZXEXP 27	NC on +3 / +2A
A27	A9	ZXEXP 30	
A28	A11	ZXEXP 32	
B1	A15	ZXEXP 2	
B2	A13	ZXEXP 6	
B3	D7	ZXEXP 1	
B4	ROM10	ZXEXP 29	+3 / +2A – NC on ALL others
B6	D0	ZXEXP 3	
B7	D1	ZXEXP 5	
B8	D2	ZXEXP 7	
B9	D6	ZXEXP 9	
B10	D5	ZXEXP 11	
B11	D3	ZXEXP 13	
B12	D4	ZXEXP 15	
B14	NMI	NMI BUTTON	
B16	MREQ	ZXEXP 17	
B17	IORQ	ZXEXP 19	
B18	RD	ZXEXP 21	
B19	WR	ZXEXP 23	
B26	A8	ZXEXP 26	
B27	A10	ZXEXP 28	

Table 3 - ZX Edge Connector to ZXEXP

Next follow the instructions for the NMI button as per the internal installation.
To assemble, things are really easy. Cut the box so that the Card Edge connector can protrude from it making sure there's enough clearance (about 3-4 mm) from the Spectrum's case (especially in the case of a +/128 etc). Cut a hole of the CF card slot and secure the reader inside the box with either screws through the holes provided on the reader's PCB or with double tape. Make holes for the Reset (if needed), NMI and upload switches and put everything together. **Done!**

Software installation

To install the software, you will need a PC - or a friend with one - and the following:

- A copy of ResiDOS 1.84 or higher from Garry Lancaster's website: **http://www.zxplus3e.plus.com/residos/**
- A program to transfer the TZX or TAP files provided to the Spectrum or tape. I use the WinTZX programs obtained by WOS

Set the upload jumper on your Spectrum (or External ZXCF box you just built) to the on position. Then type J - Symbol Shift + P - Symbol Shift + P on a regular Spectrum / + -or- LOAD "" in 128K Basic (for later Spectrums) and then ENTER.
The border will start flashing and ResiDOS will start load from either a tape you made with WinTZX or by pressing PLAY on WinTZX and feeding the signal directly from your Soundcard to your Spectrum's EAR jack.
One the software is loaded, you will be greeted by ResiDOS detection screen where the onboard memory of ZXCF will be displayed. You will then be asked to switch your upload jumper off and press ENTER. Do so and in the next question (Enable NMI for ResiDOS) answer Yes.
The Spectrum will reset and you will be shown the ResiDOS startup screen (Figure 14). Pressing the NMI button will show you the ResiDOS Task Manager (Figure 15). Read the manual online for more instructions on how to use ResiDOS. I would recommend you download either the Geoff Wearmouth's Gosh Wonderful ROM or Andrew Owen's ZX Basic SE.

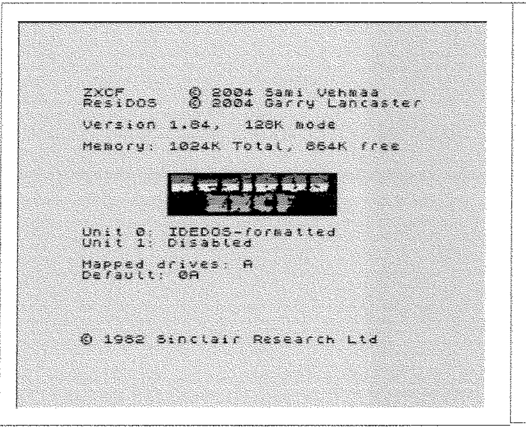
Conclusion

This article attempted to open the door to a world of upgrades available at low cost for your Sinclair computer. I do not profess to be a hardware expert (for that call Peter Graf and Zeljko Nastasic), nor an electronics magician (You should have Tony Firshman's number on speed dial!), but a mere hobbyist like most of us. If I could do it, then definitely the average user can too.

Next issue, we will deal with a fascination of mine: Sound Synthesis. We will attempt to interface a General Instruments (Motorola) AY3-8910 to a QL and Q40 in several ways. This circuit will open the door in a vast library of AY tunes available on the internet and I intend to make it work with all QL hardware under the sun. It is still a work in progress so keep your fingers crossed that I will have it ready by then! The AY chip is not only a classic sound generator, but also posseses I/O capabilities AND with fiddling the ability to replay Digital Sound.

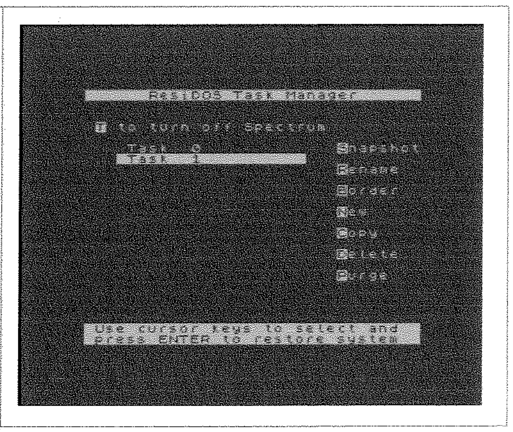
Acknowledgements - Sources

- Sami Veehma for his designs and help **http://user.tninet.se/~vjz762w/**
- Gary Lancaster for the superb ResiDOS software **http://www.zxplus3e.plus.com/residos/**
- The World of Spectrum Archive (Taper Software and Various Utilities) **http://www.worldofspectrum.org**
- Sinclair ZX Spectrum Basic Programming Manual (Greek Edition), Thessaloniki, 1983
- Sinclair ZX Spectrum +3 Technical Reference Manual Supplement, Amstrad, UK, 1987
- Mouser Electronics (for electronic parts and tools) **http://www.mouser.com/**



```
ZXCF      © 2004 Sami Vehmaa
ResiDOS   © 2004 Garry Lancaster

Version 1.84,  128K mode

Memory: 1024K Total, 864K free

Unit 0: IDEDOS-formatted
Unit 1: Disabled

Mapped drives: A
Default: 0A


© 1982 Sinclair Research Ltd
```

Figure 14 - ResiDOS Boot Screen on a 128K



```
         ResiDOS Task Manager

T to turn off Spectrum
   Task 0                   Snapshot
   Task 1                   Rename
                            Border
                            New
                            Copy
                            Delete
                            Purge

Use cursor keys to select and
press ENTER to restore system
```

Figure 15 - ResiDOS Task Manager

TURBO and Config Blocks

by George Gwilt

Introduction

Let us suppose that you are writing a program intended to be used by several different people. You want to make the program as easy to use as is reasonably possible so you arrange for a help file to be available. Of course you can't be sure where the user will store this file so your program has to cope with this uncertainty. Also, when the program detects that a key pressed by the user must be an error, you decide to indicate this by a noise. However, you realise that some people may not like that so you have to set your program to cope with either making, or not making, a noise.

There are three obvious ways of solving this problem. The first is to ask questions each time the program is started. The user will then have to say where he stores the help file and whether or not he wants a noise. This is possible, but annoying to the user.
The second way is to force the user to start the program with a parameter string containing the answers to the questions asked in the the first method. This is less annoying to the user but less easy than starting the program without a parameter string.
The third method is to provide a means of embedding the answers once and for all in the program. To do this you provide a second program, which you call "configurator". When the user runs the "configurator" he is asked the questions about the help file and noise, as in the first method but the answers are stored inside the program once and for all (or until the "configurator" is run again).
This third method was used by several early programs. For example you need a special configurator for Perfection and another one for The Editor. Unfortunately these configurators are not the same. You need to find Perfection's configurator each time you want to set a new version of Perfection. And the same is true for The Editor.

Standard Config Block

The plethora of different configurators caused the Standard Config Block to be invented. This was an excellent idea. Addition of a standard config block to a program allows it to be configured by the freely available Config or the slightly less available, but nicer, Menuconfig. One snag is that the config block is defined only for assembler programs and cannot be incorporated directly in S*BASIC programs. However, the config block is such a useful invention that, when I was chosen to update TURBO I invented the programs T_CONFIG_DATA and its companion T_COBFIG_LOAD. These allowed programs compiled by Turbo to have config blocks inserted. T_CONFIG_DATA sets up the config block and T_CONFIG_LOAD inserts the config block in compiled code.
Recently I learnt from David Gilham that there was another program BASCONFIG which also allows a config block to be set in a compiled S*BASIC program. Furthermore, there is no need for a second program to add the config block. This can be done by including the config block produced by BASCONFIG just like a set of extensions.
This knowledge caused me to produce and issue version 5.02 of TURBO which allows a config block produced by T_CONFIG_DATA to be inserted in a program during compilation by adding to it the line

```
1000 REMark %%<config block file>,1,0
```

You can, of course, as an alternative, use BASCONFIG instead of T_CONFIG_DATA to produce the config block. In that case the line inserted in the program would be

```
1000 REMark %%<config block file>,0,12
```

Because of the availability of this choice I thought it might be useful to compare the two ways of producing and adding a config block to a TURBO'd program.

Comparison of T_CONFIG_DATA and BASCONFIG

There are four areas of comparison. The first three concern the programmer and the last, the user. They are;

1. Production and alteration of the config block
2. The effect on the S*BASIC program
3. Compilation
4. The compiled program

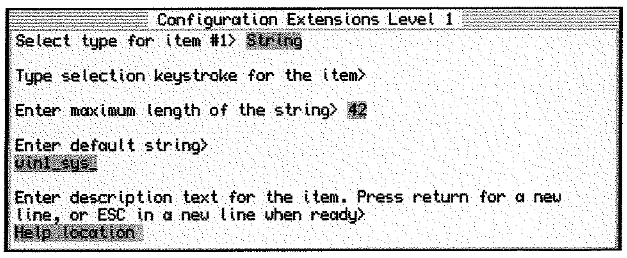
1. Production and Alteration of the Config Block

BASCONFIG leads the programmer through a set of questions very much like Config does. When the answers have all been given the config block is produced.
There is no way you can go back and alter the answer to a question once given. To do that you have to start at the beginning again - just like you have to do with Config.
To alter an existing config block you have to go all the way through all the questions again to set up a new block.

T_CONFIG_DATA is much more like Menuconfig in that the programmer can, largely, decide which question to answer next and can go back and change previous answers.

Also provision is made in T_CONFIG_DATA to allow alteration of an existing block. Perhaps this is most likely to be used for changes to the version number. The version number is held in a position near the start of the block and is displayed by Menuconfig on accessing the block so it is a good idea to keep this version number in step with the program's version number.

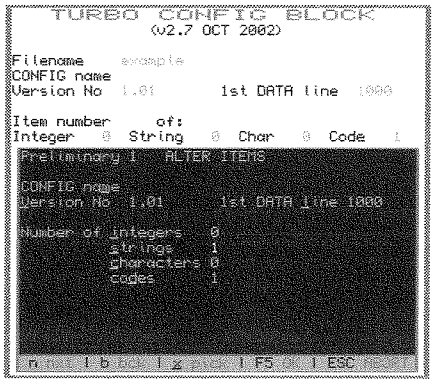
Fig. 1 shows the window used by BASCONFIG. As you answer each question, the screen scrolls up for the next question.



```
Configuration Extensions Level 1
Select type for item #1> String

Type selection keystroke for the item>

Enter maximum length of the string> 42

Enter default string>
win1_sys_

Enter description text for the item. Press return for a new
line, or ESC in a new line when ready>
Help location
```

Fig. 1

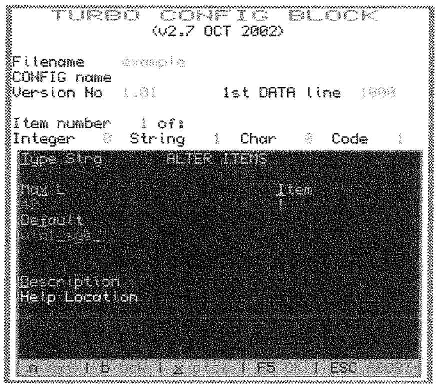
Fig. 2 shows the window of T_CONFIG_LOAD used to gather initial information. Here we have asked for one item of type string and one of type code.



```
     TURBO   CONFIG   BLOCK
          (v2.7 OCT 2002)

Filename     example
CONFIG name
Version No   1.01        1st DATA line  1000

Item number      of:
Integer   0  String   0  Char   0  Code   1
 Preliminary 1   ALTER ITEMS

 CONFIG name
 Version No  1.01        1st DATA line 1000

 Number of integers   0
           strings    1
           characters 0
           codes      1

 n nxt | b bck | x pick | F5 OK | ESC ABORT
```

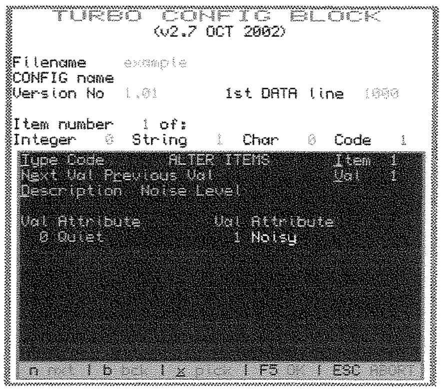
Fig. 2

Figs 3 and 4 show the windows for strings and codes.



```
     TURBO   CONFIG   BLOCK
          (v2.7 OCT 2002)

Filename     example
CONFIG name
Version No   1.01        1st DATA line  1000

Item number   1 of:
Integer   0  String   1  Char   0  Code   1
 Type Strg          ALTER ITEMS

 Max L                          Item
 42                             1
 Default
 win1_sys_


 Description
 Help Location

 n nxt | b bck | x pick | F5 OK | ESC ABORT
```

Fig. 3



```
     TURBO   CONFIG   BLOCK
          (v2.7 OCT 2002)

Filename     example
CONFIG name
Version No   1.01        1st DATA line  1000

Item number   1 of:
Integer   0  String   1  Char   0  Code   1
 Type Code         ALTER ITEMS        Item  1
 Next Val Previous Val                Val   1
 Description  Noise Level

 Val Attribute          Val Attribute
   0 Quiet                1 Noisy

 n nxt | b bck | x pick | F5 OK | ESC ABORT
```

Fig. 4

BASCONFIG provides for all varieties of config block but T_CONFIG_DATA is more limited. The limits are:

1. Level 1 config only.

2. Types long word and selection not supported.

3. String attribute 0 only. There is no option to strip spaces or designate strings as directory, filename or extension.

If there were a demand for it all these restrictions except for long word could be removed in a future version of T_CONFIG_DATA.

2. Effect on the S*BASIC program

For both BASCONFIG and T_CONFIG_DATA the inclusion requires a line such as:

```
1000 REMark %%<name> . . .
```

JUST WORDS!

IN THE SIN BIN?

Just Words! has moved. Our new contact details are given below.

Note we are now located in the Derby district of "Sinfin".

We like playing with words and just a simple typing error can turn "Sinfin" into "Sin Bin"

A different typing error, and in this case an easily made one, and "Sinfin" becomes "Sin fun".

You may choose which you prefer.

Geoff Wicks, 5b Wordsworth Avenue, Sinfin, Derby, DE24 9HQ, U.K.

Tel: +44 (0)1332 - 271366 *email: gwicks@beeb.net*

Web: http://members.lycos.co.uk/geoffwicks/justwords.htm

To access the configured items BASCONFIG uses machine code functions which form part of the config block file. All the functions start "C_" and take one parameter which is the number of the item. Thus a string might be accessed by:

```
2000 book$=C_STRG$(2)
```

The configured items in a config block produced by T_CONFIG_DATA are accessed by RESTORE and READ. Items of type word, character or code are all allotted DATA lines. Each of these is followed by a REMark containing the description of the item from the config block. Strings are slightly different. Each one, for a technical reason, requires its line to be RESTOREd and then has to be read twice. For example the string at line 1004, say, is accessed by

```
2000 RESTORE 1004
2010 READ book$,book$
```

The DATA lines themselves are produced by T_CONFIG_DATA in a file with tail _DTA. This has to be merged with the S*BASIC program.
Because a config block produced by T_CONFIG_DATA sets its items as DATA items in the S*BASIC program, it cannot accommodate a long word item. Nor does it allow the type "selection code". These are available however, in BASCONFIG by the keywords C_LONG and C_SEL$.
BASCONFIG allows the code word for strings to indicate the type of string and whether spaces are to be stripped. T_CONFIG_DATA always sets the code word to zero.

A comparison of all the types is given here:

Type	Definition	BASCONFIG	T_CONFIG_DATA	
		Function	Type of DATA	
0	string	C_STRG$(n)	string	(string)
2	character	C_CHAR(n)	word	(character)
4	code	C_CODE(n)	word	(code)
6	selection	C_SEL$	-	
8	byte	C_BYTE	word	(integer)
10	word	C_WORD	word	(integer)
12	long word	C_LONG	-	

Because S*BASIC does not differentiate between bytes and words, the DATA items for both are words. The words in brackets show the four definitions of types used by T_CONFIG_DATA and shown in Fig. 2.

3. Compilation

BASCONFIG
The config block must be LRESPRd first. The reason for this is that unless the special keywords C_STRG$ etc are inside S*BASIC's Name List the program will not compile or will compile wrongly. Thus C_SRTG$(2) will be interpreted as the third character in a simple string instead of a machine code function returning a string. Worse still, C_BYTE(2), for example, would give rise to an error, since TURBO will assume that C_BYTE is an undefined array.
When the config block has been LRESPRd the program is compiled by, say, CHARGE in the normal way.

T_CONFIG_DATA
The program is simnply compiled without any special action.

4. Effect on the User
The user will find no difference in the configuring of a program whether the block is produced by BASCONFIG or by T_CONFIG_DATA.
However, the user will find a program with BASCONFIG's config block larger and slower than one with T_CONFIG_DATA's block.

Technical Details

T_CONFIG_DATA
It is interesting that the two ways of adding a config block to a S*BASIC program should be so different. The real problem to be solved was of course how to access the configurable items. The brilliant idea of using machine code functions to access the items never crossed my mind when I dreamt up T_CONFIG_DATA. Anyway I couldn't have used the method because I had not yet at that time allowed for the inclusion of extensions in TURBO which is essential when using BASCONFIG. To solve the problem I decided to use TURBO's method of storing items of DATA. The program T_CONFIG_DATA produces both the config block itself and also the DATA lines to be used in the S*BASIC program. These DATA lines start with (for example):

```
1000 DATA "$'#*","parser
 !!!!!!!!!!!!!!!!!!!!!!!!!!! . . !!"
```

This causes the marker "$`#*" to appear in the compiled program to show where the config block must be inserted. The second DATA item starting "parser . ." effectively reserves just the space needed for the config block, excluding the items themselves. These items will be inserted in following DATA lines, also part of the _DTA file. Thus strings will appear in the _DTA file as:

```
1030 DATA "XX","000000000000000000000000"
1032 DATA "XX","11111111111111111"
```

and so on. The number of 0's, 1's etc reserve space for the maximum string defined in the config block. The "XX" data is the place where the maximum length appears.

Other items appear as (for example)

```
1034 DATA -4444:REMark Default Dataspace
```

When the config block is added to the compiled program the items "XX","0000.." at lines 1030 to 1034 will be replaced by the items in the config block.

BASCONFIG
In BASCONFIG the method of access to configured items is by means of machine code functions. The code for these functions is built into the file containing the config block itself. When the config block file is included in the compiled program the config block plus the machine code functions accessing the items are both set inside the compiled program. This is what makes the compiled program larger than if T_CONFIG_DATA has been used.

The time taken to set up and use a machine code routine is considerably greater than that used to find an item of DATA. This is what makes the BASCONFIG compiled program slower than one using T_CONFIG_DATA.

Conclusion

The difference between BASCONFIG and T_CONFIG_DATA can be summarised as:

BASCONFIG	**T_CONFIG_LOAD**
Serial operation	Operated by menu
Both config levels	Level 1 only
All types	Not long word or selection
All string attributes	Only attribute 0
Cannot alter existing block	Can alter existing block
Access item by C_xxx	Access by RESTORE and READ

I have suggested that config blocks are the best way of solving users' differing demands in one program. In practice many programs employ the first two methods mentioned at the start too with the slight amendment that the user is not forced to answer questions.

Thus a program might have:

1. A config block
2. Allow a parameter string to set defaults (overriding 1)
3. Allow alteration of items while a program is running (by loose items)

An example of a program's window using 3 is given in Fig. 5. The directory can be changed by clicking on Dir and the noise can be set on or off by clicking on Sound.

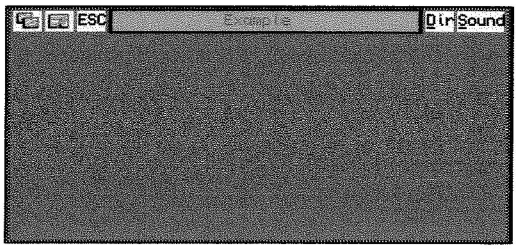


Fig. 5

EIDELWEISS: A Game for Lovers

by Stephen Poole

In the early nineties, ARK Ltd sponsored a graphics competition. For me this was quite some challenge, as I was not used to writing programs for a deadline. After much thought, I decided to write a program that would also be a game. But I did not want to do a shootem_up game, and thought that it would be interesting to do one that children could enjoy, and girls too...so I hit upon the idea of the dandelion puff: She loves me, she loves me not.
Having just returned from a trekking holiday in the French Alps, my head was full of views of summits, so I set about simulating mountain scenery, having read that such panoramas were fractal. No complex imaginary maths here though, just randomly generated overlapping triangles, with distance and atmospherics both fudged by using the RECOL command repeatedly. Don't worry if the screen goes blank, it's just fog or nightfall! Next I had to add foreground hummocks dotted of course with eidelweiss flowers. Sadly there was no space to add the sound of music!
Then came the game: A ring of petals are drawn, with the messages 'You love me', 'You love me not' printed in turn. To get the answer, instead of puffing, hit the space key: Either you go to heaven or you fall into oblivion. And the game starts again. (Eternal life)! You may have to hit the 'puff' key several times, as the routine contains code to stop you cheating, but with training, you should improve your performance.
When the peaks are drawn, after each RECOLour, the QL beeps and if you then hit the 's' (s_ave) key, you can save a view to disk. Around one screen out of six is sufficiently realistic to merit saving...There are thousands of mountains in the World: Will the QL draw known scenery?
Remember to modify the 'ct' variable if you have already saved screens. Also adjust the 'j1' and 'j2' variables as indicated in the REMarks at the head of the listing, to adapt the puffing loop to program-speed on different systems. Similarly, type SLUG (n) at the keyboard as indicated too for SuperGoldCard systems and above. Set QPC=0 if you are not using a PC. As I cannot remove my SuperGoldCard for technical reasons, I cannot tell you how to adapt j1 and j2 for unexpanded QL's. Just experiment until it feels right.
Unfortunately, QPC2 cannot do RECOLouring, so the program just pauses until the puffing begins. Also, you cannot 'LBYTE' RECOLoured screens saved on black-box QL's onto PC's, as the window addresses and dimensions are not compatible. (Advantage to original QL's for compatibility and functionality).
Originally, the Ark competition rules drastically limited the number of code lines to 50, and this program was therefor horribly crammed up, with FOR loops nested higgledy piggledy. This is why most variables are one letter long and reused throughout the program. So now I have since extensively REMarked the code and spread it out over many more lines to make it more readable. Style was also considered a winning factor, which I chose to ignore in favour of thoroughness. I do not know who won the contest, as neither ARK nor QL World ever mentionned it again. But I enjoyed the challenge and hope you like the output, which is exactly the same as with

the original program, (which was lost on a microdrive tape). Do copy your tapes and disks onto freshly reformatted media every 5 years to avoid such mishaps. (Don't let your media lose their magnetic remanence). Luckily I had hand-written the program out onto paper to get a global view of the listing. To help, I now also set WINDOW#2,360,206,0,0 which corresponds to Quanta page-width. (After 21 years without hard copy, I can finally use the printer sold with my PC thanks to QPC Print).
The most difficult part of the programming was not getting ideas for routines or debugging them, but spending hours and hours adapting the multitude of random-ranges to get output that is artisticaly acceptable. If you don't believe me, then try modifying random values without looking at the original listing and then try to get decent output again. Even as it is, output is sometimes unacceptable until RECOLoured. If colours appear gaudy, try running the program on a monochrome monitor, as the results are much better in grey-scales. As PC's do not have RECOL, the program is not half the beast that it is on ordinary Sinclair QL's. Did any of you send entries to ARK? If so I would like to see them printed, as QL World only published one program from previous competitions, called 'PLANT LIFE', by Colin Bates. It's a pity QL magazines don't set such challenges more often, as they are very stimulating when they cover seldomly-visited subjects. Perhaps I can dig out a few more programs from my hand-written archives!
Editor's comment: Steve also sent us a monochrome version after he read Geoff Wick's 'Hi-Colours' article ... which indeed looks much better. So we decided to print the monochrome version only.

```
100 REMark EIDELWEISS_bas
110 REMark by S.Poole, 1990, v21mar06.
120 CLEAR: WINDOW 512,256,0,0: j1=360: j2=540: ct=1: QPC=1
130 MODE 4: i=236: m=75: SCALE 100,-75,-50: PENDOWN
140 :
150 REMark Type Speed at keyboard before running:
160 REMark for 2.8 QPC2: SLUG 10: j1=360: j2=540
170 REMark for SGC SMSQ: SLUG 3: j1=60 : j2=90
180 REMark for SGC QDOS: no SLUG: j1=19 : j2=39
190 REMark for 128 QDOS: reduce j1 and j2 values by experiment
200 :
210 REMark Change 'ct' to current disk '_pic' number.
220 REMark Hit 's' to save _pic when QL beeps.
230 REMark Hit SPACE key to blow petals.
240 REMark (Game far best on monochrome QL monitor).
250 :
260 REPeat loop
270 :
280 REMark Fill the night sky with stars:
290 WM_PAPER 0: OVER 0: CLS
300 FOR f=1 TO RND(33 TO 99)
310  a=RND(4 TO 508): d=RND(4 TO 252)
320  e=RND(768 TO 1023): WM_BLOCK 2,2,a,d,e
330 END FOR f: CURSOR 0,i
340 :
350 REMark Rise up onto a foreground summit:
360 FOR f=1 TO m
365  rr1=RND(768 TO 1023): rr2=RND(3)
370  WM_PAPER rr1: h=RND(5): SCROLL -h
380 END FOR f: p=100: a=55: b=-a: c=35
390 :
400 REMark Draw distant mountains:
410 d=25: e=75: h=30: OVER -1
420 FOR j=1 TO RND(5 TO 9)
430  x=RND(b TO a): y=RND(a)
440  :
450  REMark Draw scree-slopes and avalanche-channels:
460  FOR f=1 TO RND(35 TO a)
470   WM_INK RND(768 TO 1023): v=RND(-c TO c)+x
480   g=RND(d TO e): w=y-g: k=v+RND(h)
490   FILL 1: LINE x,y TO v,w TO k,w TO x,y: FILL 0
```

```
500  END FOR f
510 END FOR j: OVER 0
520 :
530 REMark Draw foreground hummocks:
540 a=105: b=-a: e=25
550 FOR f=-180 TO -170 STEP RND(1 TO 3)
560  rr1=RND(768 TO 1023): rr2=RND(2): WM_INK rr1: c=RND(b TO a)
570  FILL 1: CIRCLE c,f,ABS(f)-RND(d TO e): FILL 0
580 END FOR f
590 :
600 REMark Draw eidelweiss flowers on the summit:
610 a=7: b=75: c=-a: d=1.5
620 FOR j=1 TO RND(5 TO 19)
630  L=RND(5 TO 9): x=RND(-b TO b): y=RND(-50 TO -31)
640  rr1=RND(768 TO 1023): rr2=RND(768 TO 1023): INK rr1,rr2,3
650  :
660  REMark Dont forget the numerous petals:
670  FOR f=1 TO L
680   v=x+RND(c TO a): w=y+RND(c TO a): n=RND(d TO 3)
690   e=PI/n: g=e*-1: FILL 1
700   ARC x,y TO v,w,e , x,y TO v,w,g: FILL 0
710  END FOR f: WM_INK RND(7): CIRCLE x,y,1
720 END FOR j: : u=-40: q=150
730 :
740 REMark Draw mist-layers and ambiant effects:
750 s=24: g=45: L=40: IF QPC: GO TO 830
760 FOR f=1 TO RND(2 TO 5)
770  a=RND(7): b=RND(7): c=RND(7): d=RND(7): e=RND(7)
780  RECOL a,b,c,d,e,RND(7),RND(7),RND(7): i$=INKEY$(#1,p)
790  BEEP 1234,5: i$=INKEY$(#1,200): so=i$=='s'
800  f$='flp1_Hi'&ct&'_pic'
810  IF so: SBYTES f$,131072,32767: ct=ct+1
820 END FOR f
830 :
840 REMark DANDELION-PUFF
850 j=RND(j1 TO j2): k=360/j: z=9: STRIP 0: INK 7: OVER 0
860 FILL 1: CIRCLE 0,u,z/2: FILL 0: ss=55
870 CSIZE 0,0: q$='YOU LOVE ME': r$=' NOT': s$=q$&r$
880 FOR f=1 TO j: LINE 0,u: TURNTO f*k: MOVE z: END FOR f
890 FOR f=2 TO j STEP RND(1 TO 8)
900  OVER 0: AT s,ss: CLS 4
910  IF RND(0,1): IF KEYROW(1)=64: GO TO 980
920  PRINT q$: i$=INKEY$(#1,5): AT s,ss: CLS 4: t=-1: h=f*k
930  IF RND(0,1): IF KEYROW(1)=64: GO TO 1040
940  PRINT s$: i$=INKEY$(#1,5): OVER t
950  LINE 0,u: TURNTO h: MOVE z
960 END FOR f: GO TO 1040
970 :
980 REMark HEAVENLY BLISS
990 FOR f=1 TO q STEP 2: CIRCLE g,L,f: BEEP 32000,q-f: i$=INKEY$(#1,1)
1000 FILL 1: CIRCLE g,L,25: CURSOR g,L,-65,0: OVER 0
1010 INK 0: PAPER 7: PRINT ' PARADISE! ': i$=INKEY$(#1,333)
1020 NEXT loop
1030 :
1040 REMark ABYSS OF OBLIVION
1050 a=16: b=32000: CSIZE 3,1: c$=CHR$(158): d=.75
1060 FOR f=a TO 240 STEP a
1070  AT 9,17: PRINT c$: BEEP b,f: j=(f/a)*d
1080  i$=INKEY$(#1,3): PAPER 0,RND(2),1: CLS
1090 END FOR f: i$=INKEY$(#1,100)
1100 END REPeat loop
1110 ::
```

Easel Graphics

by Dilwyn Jones

This article sets out to describe the format of Easel graphics files. It's been deduced from examining sample files created and I had no documentation to go on, so it's offered in good faith without real means of proving its accuracy.
Graphics are saved from Easel using the Print command. Option S within this command saves the graphic to a file rather than printing it to a printer. Standard QL Easel and Xchange Easel have different facilities at this point.

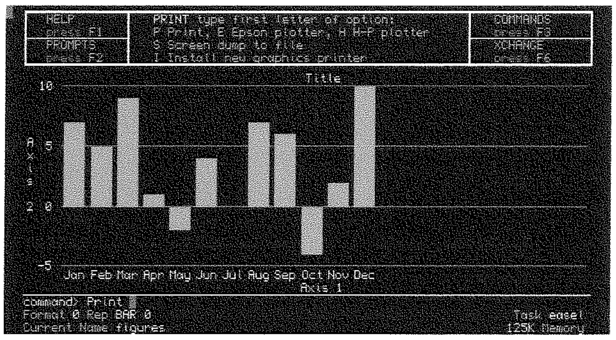
QL EASEL
This saves the screen as a standard screen in the current mode, but only works in QL mode 4. The command saves a standard QL 512x256 mode 4 screen of length 32,768 bytes.

XCHANGE EASEL
The Xchange version of Easel can also generate a screen with a 10 byte header to make it into a pointer environment area save file.
The format saved from Xchange Easel depends on the filename extension you give it.
* If you type in a filename ending with _SCR or _PIC Easel saves the graphic as a standard QL screen, of length 32,768 bytes.
* If you type in a filename ending with _CUT it generates a file which is the same as a pointer environment area save file, which has an extra 10 bytes at the beginning (before the graphic itself) as follows:

1 word	- hex value 4AFC (decimal 19196)
1 word	- width in pixels (512)
1 word	- height in pixels (256)
1 word	- number of bytes from start of one line to the start of the next line down (128 bytes)
1 byte	- mode number (0 for mode 4)
1 byte	- unused byte (normal value 0)

This only seems to work correctly in mode 4. When I try it using Xchange in mode 32 or mode 16 in QPC2, it seems to generate a CUT file with a mode number of 16 or 32, but seems to insert 32K of incorrect graphics, presumably because Xchange was written before the new high colour modes came into being and doesn't understand them. The file length of a CUT file is always 32778 bytes, irrespective of colour mode.
The extensions SCR and PIC stand for screen and picture respectively, while CUT implies this is a cut and paste file. The pointer environment area save files are indeed used as cut and paste files by several graphics programs.
We are used to PIC files being what Easel regards as CUT files, so this can be a little confusing!

RELOADING EASEL GRAPHICS
Most QL graphics programs can load Easel SCR or PIC files - anything which can load a 32K QL screen should be able to load one of these. On an original Sinclair QL it can be loaded direct to the screen with the command LBYTES filename_scr,131072 or (if you have SMSQ/E) LBYTES filename_scr,SCR_BASE if you are in a 4 colour mode 512 pixels wide.
CUT files can be reloaded into most programs which understand pointer environment area save files. The only thing to beware of is that such programs normally expect these files to have PIC filename extensions, not CUT. Most programs will allow you to change the extension.


```
HELP      PRINT type first letter of option:            COMMANDS
press F1  P Print, E Epson plotter, H H-P plotter       press F3
PROMPTS   S Screen dump to file                         XCHANGE
press F2  I Install new graphics printer                press F6
```

```
command> Print
Format 0 Rep BAR 0                                     Task easel
Current Name figures                                  125K Memory
```
Figure 1 - The Print command screen in Xchange Easel

Byts of Wood

by Roy Wood

What you see is - well a little fuzzy

I recently decided to make a unilateral land grab for a bit more desk space here at Q Branch Towers and got a TFT screen to replace my hulking great 19" CRT. The results in the desk comfort department were generally pretty good and I am glad of the extra room to stand the obligatory glass of red wine.

On the PC side all is fine and the screen is very nice but when running QPC2 the screen is not performing so well. Now this is not a criticism of QPC2 but a function of the way that flat screens actually do their business. Most screens in the low end of the price spectrum - and the QBranch budget is firmly glued down there - work well in only a few of the available resolutions and only give average performance when asked to run in the other modes. This, of course, because the screens are comprised of discrete pixels as opposed to scanned lines as on a CRT monitor. My screen works well in 1280 x 1024 mode which is the resolution I use for the PC and imperfectly in 1024 x 768 which is the one I use for QPC2. There is also a, somewhat annoying, pause when switching between the two. This is not just a QPC2 problem. Gamers will also notice a drop off when playing games which switch the modes. The solution, of course, is to move QPC2 to a higher resolution. Although this is a fairly trivial matter of reconfiguring the startup screen for QPC2 it does leave you with small text and a slight problem with reconfiguring the QDT desktop. I will come on to that a little later on a bit about monitors.

I Rename This Column the Serendipity Time....

Well I don't really, because that would be even sillier that the name i gave it when I first started writing it, but I am amazed that, when ever I start to write something for this other people contribute - even with their knowledge. As I was running through the things I wanted to say about the subject of monitors The QL-Developers list sprung into life and started talking about the same thing. Admittedly the QL-Developers list is more about LINUX on the Q40/Q60 than it is about QL subjects but the discussion there mirrors some of the things which I am saying here.
The discussion started with Wolfgang Mühlegger talking abut Linux but then he asked if it was not possible to get a TFT to work with the Q60. Thierry Godefroy replied (and I will quote him here for those of you who do not subscribe to the QL-Developers list):
'*Alas, the Q60 display is using non-square pixels (like the QL did), and the LCD screens can only deal with square pixels unlike what happens on a CRT monitor where the geometry of the pixel is only determined by the horizontal and vertical resolution. At best, a LCD screen will display the Q60 screen on part of its surface (1024x512 instead of 1024x768) and the picture will appear "distorted" (not high enough), making the circles into ellipse, etc... At worst, the LCD monitor will try and interpolate the missing lines to force a 1024x768 display, and the picture will be very ugly (the text will be unreadable).*
The only solution would be to implement a new resolution (800x600 would be fine and would fit the video memory as it got 10% less pixels than 1024x512), but this would involve changing the glue logic chips on the Q60.
Too bad the Q40/Q60 did not adopt a square pixel solution from the very start, like the QXL, QPC and the Aurora did...'
As usual with Thierry it was a very concise explanation of the problems involved.
Peter Graf replied to this commenting that
'*The problem would be much smaller if TFT were multisync like CRT. Technically, multisync wouldn't be very hard to implement in a TFT screen, because it has adjustable interpolation units for x and y anyway. Unfortunately the industry has decided to ignore all but some fixed resolutions, and the adjustment ranges are usually too small for our needs.*
Every TFT screen can interpolate missing lines for traditional PC resolutions (and even non-square pixels in some cases). The results are often not that bad. So the problem is not so much the need for interpolation, but that QL and Q60 resolutions are not in the monitor's fixed list. In most cases, 1024x512 is misinterpreted as 800x600 or 640x480, which often leads to two additional effects:
- The rate at which the analogue signal gets sampled (before interpolation) is too low and additionally "blurs" the picture.
- Part of the picture is outside the screen area.
I have tried to implement 800x600 in the Q60 video controller by replacing an existing resolution, but 800x600 requires more logic for internal calculations than resolutions with exponent of 2; the chip resources are too limited.
What I can offer is 1024x768, with a usable area of 1024x512, the rest is black. This works well on some (not all) TFT, but leads to elliptic circles etc. Also it has a very low vertical refresh rate, making it unsuitable for CRT.
The only satisfying solution seems to be a new Q60 graphics card. Not that I lack ideas, but it would be a very difficult project for many reasons.'
Derek Stewart then replied:
'*I do not think that this is case, as QPC and Aurora can use variable resolution aspect ratios, QXL departed from the QL display ratio of 2:1 to 4:3. The Q40/60 keeps the original QL display ratio of 2:1, which does a reasonable thing. I suppose that if the TFT monitor controller can be tricked into thinking that the 1024x512 mode of the Q60, then we may get a better display, but this might require alteration to the sync rates.*'

Sorry to sling so much quotation from the list into this but the three contributions neatly explain the problem with TFT screens. Unfortunately for us, the CRT is an endangered species. The only reason it still exists at all any more is that photographers prefer them to the 'hard' TFT displays. In all other aspects manufacturers and retailers prefer TFTs because they are cheaper to ship, take up less space to store and look very 21st Century. Low end CRTs are very hard to find and many of the major CRT manufacturers have stopped making them entirely. Worse news for lovers of these bulky objects comes in the form of the new NEC range of flat screens which, it claims, perform as well as high end CRT screens. Maybe this means they are more adjustable and would work well with the Q60 but, at £800 + for the lowest spec one, I cannot see too many people taking a gamble on it. The cheaper end of the TFT market, which is where many of us will be lurking, does not indulge in the kind of electronics that make all resolutions equally look good. If I reset my PC to 1024 x 768 it looks just as patchy as the QPC 2 display does and, if I reset the QPC2 resolution to 1280 x 1024 I get the same crisp display I have on the PC.
Technology would seem to be leaving us behind but some of the reactions to that would seem to be a little short on the reality check scale. Of course it is feasible that TFT screens could be made to multisynch but why bother? Most PC users use the thing as it comes out of the box. If they realise that there are settings you can change they are convinced it needs an expert to do it. Therefore, by its own popularity is it undone. If we do not get the level of sophistication we think we need or deserve it is because, like television and music, it is designed for a mass market and those with IQs higher than their shoe sizes are not catered for.

Blow It Up

Given, therefore, that many of you may be getting into that same situation I thought I had better explain how to make a new background screen for your QDT desktop. You will need two QL-programs and a Windows / Linux based program to create the original image and Marcel Kilgus' sprite converter. The QL programs and the sprite converter are all freeware and can be downloaded from various QL sites.
First generate a PNG file of the final screen size. This has to be the exact screen size that you want the final file to be.
Next, in Windows, run Marcel's sprite converter to convert to a sprite 8 bit : Mode 16 : Aurora 16 bit : Mode 32 : QPC2
Once this has been converted move onto the QL-platform and exec snatch4_obj (available from Dilwyn Jones and set saving name and location.
Once that is running in the background Irun loadspr_bas to load sprite to desktop. If this is displayed how you want it press CTRL S to capture it with snatch4. This will then create the background image you can use for QDT. You do not have to be running QDT for this to be used, however because the facility is there in the later versions of SMSQ/E.
Most of the programs I have mentioned here can be obtained from either Dilwyn Jones or Marcel Kilgus' websites.
As I wrote this I received notice that Dilwyn had just released a program called BMP which can be found on his website. I have not had too much time to play with this yet but it does seem to be a useful utility. It is always nice when people put this amount of time and effort into producing software and then giving it away for free - especially when there is very little commercial software available. If you can get the picture you want into the correct size for the screen size you are using then you can use Dilwyn's program to do the whole conversion in one go.
One suggestion for Dilwyn might be to add a resize option to the program but that may be a very hard option to add. I am not sure how complex it is to resize graphics. If it could be added it would be a neat way to make you own QDT or SMSQ/E background screen. Just choose a PC graphic file and a screen size and convert it.

Graphics and The QL

It is nice to see a genuine graphics program being written for the QL. One of the formats that it converts to is Page Designer 3. This is a program which has had a chequered history. The original was, I think, part of Dilwyns portfolio and then the onus on writing and improving it fell to Barry Ansell who did a very good job of converting it to run under SMSQ/E but quit before the colour driver were released. The program hangs in limbo these days. Q Branch was involved in selling it for a while but I was told to destroy any copies I had left to sell when Barry Quit the QL scene. At the time I was trying to get him to hand it over to Rich Mellor for further development but he would not allow this.
In many ways the QL lacks a definitive Graphics program. Page Designer 3 had some very good features and some very odd ones. My own preferred program was always LINEDesign because it was a very powerful tool. It lacks the

sophisticated manipulation facilities offered by PC programs like Adobe's Illustrator but it was a very quick and easy program to use and the results could be spectacular.
This is a program which is now, apparently, open source so a good C programmer could actually develop this further, adding colour and a better support for other file formats. Maybe we could get Dilwyns BMP program to covert into and out of LINEDesign format. That would have been a useful addition too.

DATA in old Formats

There was a recent letter published in the Guardian Newspaper's 'Ask Jack' Column in which someone asked if there was any software capable of reading files in an old and no longer supported PC database format of course the answer was that there was no and no one made filters that would allow users to extract that data into modern formats. The reply said 'Never put Data into a format that you cannot extract it from'. All very well but how do you know which formats are going to go the way of the Dodo and which are going to carry on into posterity and be hailed as a masterpiece of modern software design? Worse still, of course, some of the more popular software formats are, in fact, several lightyears from the masterpiece accolade. This I suppose, is only to be expected given the way that modern commercial software is produced but how can you make am educated guess about what constitutes a safe bet?
One answer many of you may give is to store it in Quill or Archive (although Archive's drastically bad design has destroyed more data than it has saved I have no doubt.) I suppose the answer to all this would be a universal Data format but then we have all been down that road with the 'universal graphics formats' many of which proved to be anything but that. We should count ourselves lucky that although we have a 22 year old system many of our older formats are so well supported.

Birthday Parties?

Almost as an aside to the above I was reading in the Guardian's technology section that it is 25 years ago this year that IBM produced their first PC (an acronym that stood for Plug Compatible, I believe, and not Personal Computer as most people today think.) I was wondering where all the 'PC is 25' parties are? I suppose the users of these machines back then were fairly excited in a corporate kind of way (wow, we can calculate the interest on this loan much faster now and it only cost us $1500.00 dollars) but it was never a real home computer in the way that we know it today. Mind you neither was this the intention behind the QL. Somehow or other that long black box with its malfunctioning microdrives stirred something in the hearts of those who took to it despite the bad publicity and botched launch. There may be a few less of us now than there were back in the late 1980's but there are still black QLs in use out there which is more than can be said for IBM's creation.
I was also told by a colleague of mine that IBM were not the first people in Europe to sell a 'Home Computer'. Apparently Olivetti beat them to the European market by a few months. Not, of course that it did them any good. No one recalls the Olivetti Home computer now and I doubt if anyone even remembers that they ever made one. They had a brief stab at printers a while ago but they were not that good either.

SPAM SPAM SPAM SPAM

Just for a little light humour and a general large dose of incredulity at what people could fall for I feel I have to pass on this wonderful piece of spam I received at work.
I got notification that I had won one million Euros in a Microsoft Promotional Drawer. Now I realise that this may be no surprise to many of you since I am sure that your mailboxes are all heaving with emails saying you have been successful in winning similar awards but this one takes a large biscuit because, at the bottom, it give the email address I have to contact as '**microsoftclaim20@netscape.com**'. Now the concept that Microsoft would ever organize a promotional handout and not use the Microsoft.com address is fairly extreme but to use a Netscape address...... well!
A friend of mine, who works for the computer forensics department of Scotland Yard, told me he had a woman complaining that her computer had been 'hacked' and her bank and credit card details stolen. She had lost several thousand pounds. When he examined her computer he found an email notifying her that she had won a similar lottery in Luxembourg and a later exchange of mail in which she was asked for her bank details, credit cards and PINs to facilitate the transfers. When he asked her about this she said she had provided the details and was waiting for the money. He asked if she had ever entered a lottery in Luxembourg and she said 'No, but it looked genuine'. I suppose the only thing that surpasses human gullibility is human greed.

The QL Show Agenda

Regular QL Meeting - (NL) Eindhoven

Saturday, 14th of October, 10:00 to 16:00

Pleincollege St. Joris, Roostenlaan 296

JMS will be there - with his usual range of QL/SMSQ programs, special offers, CD-Rs...

The North American QL Show 2006

The North American QL Show will be held on September 30th in Niagara Falls, Ontario, Canada. We will meet in the lobby of the Super 8 Motel for a group dinner at a nearby restaurant at 6pm Friday September 29th. The show will be in the Super 8 Motel conference room from 10am till 4pm on Saturday. A simple free lunch will be provided during the show.

The foliage will be starting to change in late September so there will be lots of colors to enjoy in addition to Niagara Falls and casinos which are within walking distance of the motel. Visitors from distance places can fly into Toronto or Montreal and stay within Canada the whole time if they wish. They can also fly into many US cities. The Canadian $ is currently about 0.9 US$.

So far Jim Hunkins, Roy Woods, Tony Firshman, Al and Dorothy Boehm, Bill Cable and Mary Boyle are planning to attend. Jim will give a live QDT demo and Roy will show off the latest QPC and Tony will fix whatever is broken. Please spread the word.

Location of Motel where show will be held: SUPER 8 MOTEL - Niagara Falls - Canadian side
Prices in CAN $ - QEW to Hwy 420 * 5706 Ferry St, Niagara Falls, ON L2G 1S7, CA
Phone: 888-442-6095 * Fax: 905-356-7760 * E-mail: **3386@hotel.cendant.com**
http://www.super8.com/Super8/control/

The price of a room with 2 double beds for Friday Sep 29th will be $80 and Saturday night $116. When reserving specify that you are with the QL Computer Club and eligible for the General Managers Discount. There is wireless internet at the motel. Check in time is 1pm.

Bill Cable is the contact person for any questions:
510 St Gaudens Road, Cornish, NH 03745 USA
Phone 1-603-675-2218 E-mail: cable@cyberportal.net

The Next Issue

We plan to have the next issue ready at the end of October/beginning of November. As always, it depends on how quickly we will get reviews, articles etc.

Maybe we get enough material to have it ready at the Eindhoven QL show, middle of October (please see above), otherwise at the Byfleet show, 5th of Nov. (see Quanta ad on page 19)